O^3HSC

Outsourced Online/Offline Hybrid Signcryption for Wireless Body Area Networks

Liu, Suhui; Chen, Liquan; Wang, Huaqun; Fu, Shihui; Shi, Lin

DOI
10.1109/TNSM.2022.3153485

Publication date
2022

Document Version
Final published version

Published in
IEEE Transactions on Network and Service Management

Citation (APA)
Liu, S., Chen, L., Wang, H., Fu, S., & Shi, L. (2022). O^3HSC: Outsourced Online/Offline Hybrid Signcryption for Wireless Body Area Networks. *IEEE Transactions on Network and Service Management*, *19*(3), 2421-2433. https://doi.org/10.1109/TNSM.2022.3153485

O^3HSC: Outsourced Online/Offline Hybrid Signcryption for Wireless Body Area Networks

Suhui Liu, Liquan Chen, *Senior Member, IEEE*, Huaqun Wang, Shihui Fu, and Lin Shi

***Abstract*—Wireless body area networks (WBAN) enable ubiquitous monitoring of patients, which can change the future of healthcare services overwhelmingly. As the collected data of patients usually contain sensitive information, how to collect, transfer, store and share data securely and properly has become a concerning issue. Attribute-based encryption (ABE) can achieve data confidentiality and fine-grained access control simultaneously. Identity-based ring signature (IBRS) allows patients to prove their identity without leaking any extra (private) information. However, the heavy computational burden of ABE and IBRS is intolerable for most power-limited mobile devices, which account for a large proportion of WBAN devices. This paper combines the attribute-based online/offline encryption (ABOOE) and IBRS to achieve an outsourced online/offline hybrid signcryption (O^3HSC) scheme. As far as we know, this scheme is the first signcryption scheme that adopts IBRS and satisfies online/offline signcryption simultaneously. O^3HSC divides the key generation and signcryption into offline and online phases to increase the throughput of the central authority and save the power resources of mobile devices, respectively. Besides, outsourced decryption and public signature verification are also realized. O^3HSC achieves security under CCA and CMIA, and the performance analysis shows that O^3HSC is a lightweight and applicable scheme for WBAN.**

***Index Terms*—Attribute-based online/offline encryption, identity-based ring signature, outsourced decryption, wireless body area network.**

I. Introduction

THE WIRELESS body area network (WBAN) is composed of a large number of edge devices, especially mobile devices with limited power resources. With the help of the Internet, WBAN can easily access the body sensor data and resources, no matter where the patient is or when an emergency occurs. In this way, it helps reduce the treatment costs, provide effective services, more timely reports and faster access to treatments. Despite all those convenient services, the security and privacy issue of WBAN data remains a major concern because its medical data is usually relevant to users' privacy [1]. Although traditional encryption methods can protect the confidentiality of data, they face high computational costs and inefficient data access and sharing.

The attribute-based encryption (ABE) [2], [3] was proposed to achieve data security and fine-grained access control simultaneously. Ciphertext-policy ABE (CP-ABE) [4] is more suitable for owner-specified applications than the key-policy ABE (KP-ABE) [5], [6] as the ciphertext is associated with an access structure defined by the data owner itself. In this way, only these data users whose attribute sets satisfy the ciphertext access structure can access the data. While in the KP-ABE, the user's private key associated with the access structure is uniformly issued by the attribute authority, which means that the correctness of access control depends on the credibility of the attribute authority.

The main component of WBAN is a large number of mobile devices with limited power and storage resources, such as cellphones and sensors, which means that they cannot afford high power consumption or large storage space calculations. However, the encryption and decryption of CP-ABE require numerous bilinear pairing operations and exponentiation operations. Specifically, two scenarios need to be considered. When a WBAN mobile device acts as an encryptor, a new primitive, the online/offline encryption [7], [8] was proposed, in which the encryption is divided into offline and online phases. Most of the computational work is done during the offline phase when the mobile devices are powered on. Then in the online phase, when the devices need to encrypt real data, only a little power will be consumed. The second scenario is when the mobile device acts as a data user (decryptor), the ABE with outsourced decryption (OABE) [9], [10], [11] was proposed to move the heavy decryption cost from the WBAN device to a third-party server.

For real-world applications, confidentiality alone is far from enough, especially for data-sensitive areas such as WBAN. To resist active adversaries who try to tamper the data transmitted on a public channel and reduce the waste of resources used for verification, it is essential to verify data authenticity and integrity before accepting the data. Many signature algorithms have been proposed, such as RSA, identity-based signature (IBS), attribute-based signature (ABS), etc. Recently, the ring signature (RS) [12] was designed particularly for privacy protection applications. In the ring signature, a true signer combines the information of multiple fake signers to

Manuscript received 29 October 2021; revised 17 February 2022; accepted 19 February 2022. Date of publication 23 February 2022; date of current version 12 October 2022. This research is supported by National Key Research and Development Program of China (2020YFE0200600) and National Natural Science Foundation of China (No.62002058). The associate editor coordinating the review of this article and approving it for publication was Q. Li. *(Corresponding author: Liquan Chen.)*

Suhui Liu, Liquan Chen, and Lin Shi are with the School of Cyber Science and Engineering, Southeast University, Nanjing 211102, Jiangsu, China (e-mail: suhliu@126.com; lqchen@seu.edu.cn; 230209338@seu.edu.cn).

Huaqun Wang is with the Jiangsu Key Laboratory of Big Data Security & Intelligent Processing, College of Computer, Nanjing University of Posts and Telecommunications, Nanjing 210003, China (e-mail: wanghuaqun@aliyun.com).

Shihui Fu is with the Faculty of Electrical Engineering, Mathematics and Computer Science, Delft University of Technology, 2628 CD Delft, The Netherlands (e-mail: shirhuifu@gmail.com).

Digital Object Identifier 10.1109/TNSM.2022.3153485

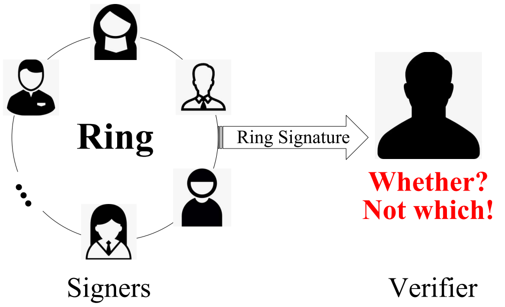


Fig. 1. Ring Signature.

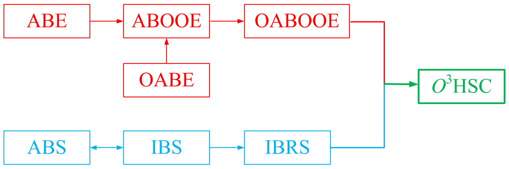


Fig. 2. Design Idea.

complete a signature. As illuminated in Figure 1, when verifying the validity of a signature, the verifier can only verify whether the signer of this signature is one of the signers in the ring, but cannot exactly deduce which signer it is. To transport or store messages of varying length in a secure and authenticated way with an expense less than that required by "signature followed by encryption", Zheng introduced a new primitive, signcryption [13]. Since signcryption (SC) can efficiently implement encryption and signature within one primitive, many extensions have been proposed, such as the identity based signcryption (IBSC) [14] and attribute based signcryption (ABSC) [15].

The existing SC schemes face two major challenges when applied to the WBAN environment, one is the heavy computational overhead, and the other is the user privacy issue. In response to the first issue, we made two improvements to the ABOOE. First, based on inheriting the original online and offline encryptions, the key generation is also changed to the online and offline mode to improve the overall efficiency of the system. Then, the decryption is outsourced to a resource-rich third party to achieve the overall lightweight of our scheme. For the second issue, we combine the outsourced ABOOE with an IBRS instead of the IBS or ABS. On the one hand, IBSs are more efficient and have a wider range of applications than ABSs. On the other hand, compared with IBSs, IBRSs can protect the privacy of user identities. In this way, we propose a new cryptographic primitive: the outsourced online/offline hybrid signcryption (O^3HSC) in this paper, which is specifically applicable for mobile-devices-dominated WBAN. The design idea is illustrated in Fig. 2.

Main contributions of O^3HSC are concluded as follows:

- *Innovative:* O^3HSC firstly combines ABOOE and IBRS into a new primitive, online/offline hybrid signcryption, which ensures the confidentiality of data effectively and efficiently while enabling the identity authentication without leaking any private information. Furthermore, the securities against chosen-ciphertext attacks and chosen-message-and-identity attacks are proved.
- *Lightweight:* O^3HSC divides both the de-signcryption key generation and signcryption into online and offline parts. Simultaneously, it achieves outsourced de-signcryption. Compared with the traditional ABE schemes, our lightweight O^3HSC scheme is more suitable for resource-limited mobile devices in WBAN.
- *Practical:* O^3HSC not only protects the user privacy but also realizes another meaningful function, the public verification, which means anyone who gains the ciphertext can verify the authentication of data.

Section II reports the most related works. The system model and security models are described in Section III and the detailed construction of our scheme is in Section IV. After presenting the security and performance analysis in Section V and Section VI respectively, we conclude in Section VII.

II. Related Work

Since Rivest *et al.* [12] firstly proposed the ring signature scheme, many more efficient schemes have been put forward. In [16], Zhang and Kim first proposed an identity-based ring signature (IBRS) scheme which provides the anonymity of signers and simplifies the certificate management in the public key system. Then in [17], Chow *et al.* proposed a more efficient IBRS scheme that only requires two pairing operations to signcrypt regardless of the size of the user group.

Both the encryption and decryption phases of ABE require heavy computational cost as the number of exponentiation computations is usually linear in the number of attributes involved in the access structure. To deal with this overhead issue, many lightweight ABE schemes with improved encryption or outsourced decryption methods have been proposed [18]. To meet the special requirement of power-limited mobile devices, Sowjanya and Dasgupta [19] designed an ECC-based lightweight ABE, which did not consider the authentication issue. Hohenberger and Waters [8] proposed the online/offline attribute-based encryption where most of the computation work is done without knowing the encryption access structure and the message to be encrypted during the offline phase. Based on that, several schemes with online/offline mode encryption were designed in [20], [21].

On the other hand, Qin *et al.* [22] implemented a verifiable out-decryption to decrease data users' overhead by delegating most of the decryption work to an untrusted third-server, the cloud server. Liu *et al.* [23] proposed a multi-authority ABE scheme that realizes the verifiable outsourced decryption and white-box traceability. A blockchain-assisted searchable ABE scheme was proposed by Liu *et al.* [24], which utilizes the cloud to perform the outsourced decryption and the blockchain to achieve efficient user revocation, respectively. Feng *et al.* [25] proposed an ABE scheme with parallel outsourced decryption by taking advantage of edge intelligence devices.

In order to reconcile the conflict of low efficiency and the requirements of confidentiality and authenticity, signcryption schemes are getting increasing research attention. In [26],

TABLE I
COMPARISON OF FUNCTIONALITY

Scheme	Scheme Policy	Online Offline	Large University	Security Model	Signer Privacy	Public Verification	Outsourced Decryption
[30]	CPABE+IBRS	×	×	IND-sCPA + EUF-CMIA	✓	×	×
[32]	CPABOOSC	✓	✓	IND-sCCA2 + EUF-sCMA	✓	×	×
[33]	CPABSC	×	✓	IND-CPA + EUF-CMA	✓	✓	✓
[34]	CPABE+KPABS	×	✓	IND-sCCA2 + EUF-sCMA	✓	×	✓
[35]	CPABSC	×	×	IND-sCCA2 + EUF-CMA	✓	×	✓
Ours	CPOOABE+IBRS	✓	✓	IND-sCCA2 + EUF-CMIA	✓	✓	✓

Abbreviations: ✓: support functionality. ×: do not support functionality. IND-sCPA: indistinguishability against selective chosen-plaintext attacks. EUF-CMA: existential unforgeability against chosen-message-attacks. IND-sCCA2: indistinguishability against selective chosen-ciphertext attacks. EUF-CMIA: existential unforgeability against chosen-message-identity-attacks.

the formal proof of the security for the signcryption was presented for the first time. The notion of identity-based ring signcryption (IBRSC) was put forward by Huang *et al.* [27] to provide the anonymity of signcrypters. And the first attribute-based signcryption (ABSC) scheme was proposed by Gagné *et al.* [28], which achieves not only fine-grained access control but also security and confidentiality. In [29], Yu and Cao proposed an attribute-based signcryption scheme with hybrid access policy where they adopted a key-policy signature and ciphertext-policy encryption, yet the overhead of the calculations was too heavy for an IoT environment. To protect data security and user privacy in the WBAN, Wang *et al.* [30] combined an ABE scheme and an IBRS scheme. Recently, Eltayieb *et al.* [31] combined ABSC and blockchain technology to protect data security in the cloud server, which does not consider the computational and storage overhead.

Concerning the heavy computational burden, some improved ABSC schemes were proposed. Liu *et al.* [36] proposed an RSA-based certificateless signcryption scheme to manage the WBAN data. Rao [32] proposed an online/offline attribute-based signcryption that supports large-size universes and monotone Boolean function predictions. But this scheme may leak user information during the verification, and the decryption overhead is heavy. In [33], Kibiwott *et al.* proposed a fully outsourced ABSC scheme where the tasks of key generation, signcryption, and de-signcryption are all outsourced. Recently, Yu *et al.* [34] proposed a lightweight ABSC scheme with a hybrid access policy, which achieves a low-latency outsourced verification and decryption by using location-advanced fog nodes. In [35], Belguith *et al.* proposed a privacy-protected ABSC scheme with verifiable outsourced decryption, which prevents a lazy server from returning a previously computed designcrypted ciphertext. However, all lightweight ABSC schemes mentioned above are not applicable for WBAN as none of them considers both low-power encryption and decryption.

Table I summarizes the functionality comparison of some related signcryption works. It can be seen from Table I that the existing signcryption schemes are not suitable for WBAN mobile devices with limited power supply due to their heavy overhead issues. The scheme proposed in this paper is designed for this special application scenario, which achieves power-saved signcryption and outsourced de-signcryption simultaneously. More importantly, our scheme uses attribute-based settings to achieve one-to-many data sharing and adopts identity-based ring signatures to achieve privacy-protected data authentication.

III. SYSTEM DEFINITION

A. System Model

Our outsourced online/offline signcryption (O^3HSC) scheme contains five participants: **Central Authority (CA)** is a trusted server that is responsible for generating the signing keys for controllers based on the patient identities and the de-signcryption keys based on the user attribute sets; Wearable or implanted **Sensors** are used to sense vital signs or environmental parameters. Then they transfer the data to the associated controllers; **Controllers** usually refer to mobile devices such as patients' telephones which aggregate information from sensors. Then the controllers encrypt information and send them to the cloud server. Each controller is uniquely identified by the patient's identity and can sign using a signing key distributed by the central authority; **Cloud Server (CS)** is responsible for storing patients' private health information. We assume that the CS in our system is honest but curious; **Data Users (DUs)** (such as healthcare providers, doctors, and researchers) try to access the patients' health information based on their attributes.

The basic procedure of our outsourced online/offline signcryption (O^3HSC) scheme consists of the following four phases (ten algorithms).

1. *Setup*$(1^\kappa) \rightarrow (PP, MK)$: It is performed by the CA, which takes as input the security parameter κ and generates the public parameters PP and a master key MK.

2. *Key Generation:*

2.1 *SignKeyGen*$(PP, MK, ID_i) \rightarrow (PK_{ID_i}, SK_{ID_i})$: The CA takes $(\mathrm{PP}, \mathrm{MK}, \mathrm{ID_i})$ as inputs and generates the signing key for the user based on its identity $\mathrm{ID_i}$.

2.2 *OffDscKeyGen*$(PP, MK) \rightarrow SK'_d$: This algorithm is performed by the CA, which takes (PP, MK) as input and generates the de-signcryption keys for users before knowing the specific attribute sets of users.

2.3 *OnDscKeyGen*$(SK'_d, U_{uid}) \rightarrow SK_d$: The CA performs this algorithm based on the user's attribute set $\mathrm{U_{uid}}$.

2.4 *OutDscKeyGen*$(SK_d) \rightarrow SK_{od}$: The CA runs this algorithm to generate the outsourced de-signcryption keys which are used to outsource the heavy decryption burden to the CS.

3. *Signcryption:*

3.1 *OffSigncrypt*$(PP, \mathbb{ID}, SK_{ID_i}, \{PK_{ID_i}\}_{ID_i \in \mathbb{ID}}) \rightarrow CT'$: This algorithm is performed before controllers knowing

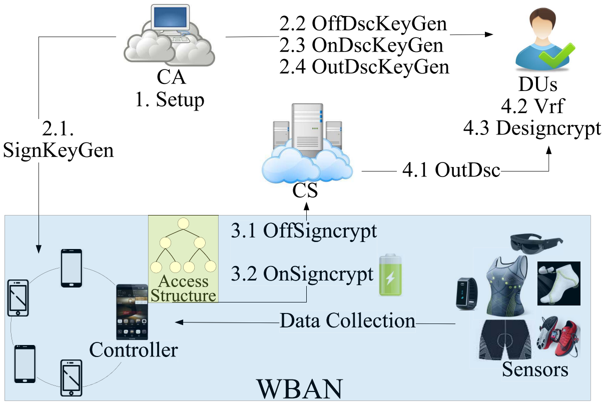


Fig. 3. System Model and Procedures.

the concrete message and the access structure when they are plugged into a power source, where PP is the public parameters, $\mathrm{SK}_{\mathrm{ID_i}}$ is the signing key, and $\mathbb{ID}$ is a set of identities.

3.2 *OnSigncrypt(PP,* $\mathbb{ID}$*,* SK_{ID_i}*,* $\{PK_{ID_i}\}_{ID_i \in \mathbb{ID}}$*,* $(\mathbb{M}, \rho_e)$*, m, CT′)* → *CT:* It is performed by the controller with input after knowing the specific message and the access structure.

4. *De-signcryption:* most of the de-signcryption work is outsourced to the CS to decrease the burden of DUs.

4.1 *OutDsc(CT,* U_{uid}*,* SK_{od}*)* → *OCT:* If the user's attribute set satisfies the access structure, the CS uses the outsourced de-signcryption key received from the user to partially decrypt the ciphertext.

4.2 *Vry(CT,* $\{PK_{ID_i}\}_{ID_i \in \mathbb{ID}}$*)* → *0/1:* Anyone in this system can use the user's public signing key to verify the signature. The authenticity of the signature is checked if the output of this algorithm is equal to 1.

4.3 *Designcrypt(OCT, CT,* SK_d*)* → *m:* The user can use the partially decrypted ciphertext and its de-signcryption key to quickly decrypt the ciphertext.

Fig. 3 exhibits procedures for implementing the proposed scheme in the WBAN. Firstly, the system setup is accomplished by a trusted server, CA, which is responsible for generating user decryption keys. Then, those end sensors in the WBAN, such as smartwatches, will generate data through monitoring persons' body conditions continuously. Next, data will be collected by its id-key-controlled controller, which performs the signcryption only when it is plugged into a power source. More importantly, the controller coalesces several other controllers to sign the data with the id-based ring signature to protect privacy. Then, the ciphertext will be sent to the CS for storage. When a data user wants to access the person's data, the CS will out-decrypt the ciphertext with the user's out-dec key first. Finally, the data user can verify the data source and decrypt it efficiently.

B. Security Model

1) Message Confidentiality: The message confidentiality security model of an (O^3HSC) scheme is defined through the indistinguishability of ciphertexts against selective encryption structure and adaptive chosen ciphertext attack (IND-sES-CCA2). Details of the game $\mathrm{Game}_{\mathbb{A}}^{\mathrm{IND-sES-CCA2}}(\kappa)$ between a challenger $\mathbb{C}$ and a PPT adversary $\mathbb{A}$ is presented as follows.

Initialization: The challenger $\mathbb{C}$ specifies the attribute universe U. The adversary $\mathbb{A}$ submits an encryption access structure $(\mathbb{M}^*, \rho_e^*)$ which is used to generate the challenge ciphertext during the Challenge phase.

Set Up: $\mathbb{C}$ runs the $\mathsf{Setup}(1^\kappa)$ algorithm to generate the public parameters PP and the master key MK. $\mathbb{C}$ sends PP to $\mathbb{A}$ and keeps MK secret.

Query I: $\mathbb{A}$ issues adaptively a polynomially bounded number of the following queries. $\mathbb{C}$ initializes an empty table $\mathbb{T}$ and an empty set $\mathbb{S}$.

1) *Signing Key:* $\mathbb{A}$ submits an identity $\mathrm{ID_i}$ to $\mathbb{C}$, then $\mathbb{C}$ runs the SignKeyGen algorithm and returns $\mathrm{SK_{ID_i}}$ to $\mathbb{A}$.

2) *De-signcryption Key and Outsourced De-signcryption Key:* $\mathbb{A}$ submits an attribute set $\mathrm{U_{uid}}$ which does not satisfy the challenge access structure $(\mathbb{M}^*, \rho_e^*)$. $\mathbb{C}$ sets $\mathbb{S} = \mathbb{S} \cup U_{uid}$, then it runs the OffDscKeyGen(PP,MK) algorithm and the $\mathsf{OnDscKeyGen}(\mathsf{SK}'_\mathsf{d}, \mathsf{U_{uid}})$ algorithm to generate the de-signcryption key $\mathrm{SK_d}$. Next, $\mathbb{C}$ first searches the table to answer the outsourced de-signcryption key queries. If this entry exists, it returns $\mathrm{SK_{od}}$ directly. Else, it runs the $\mathsf{OutDscKeyGen}(\mathsf{SK_d})$ algorithm to generate the outsourced de-signcryption key and stores the entry $(\mathrm{U_{uid}}, \mathrm{SK_d}, \mathrm{SK_{od}}, \mathrm{OVK} = \mathrm{z})$ in the table $\mathbb{T}$. Finally, $\mathbb{C}$ returns $(\mathrm{SK_d}, \mathrm{SK_{od}})$ to $\mathbb{A}$.

3) *Signcryption:* $\mathbb{A}$ submits a message $m \in \mathrm{M}$, an identity set $\mathbb{ID}$, and an encryption structure $(\mathbb{M}, \rho_\mathrm{e})$. $\mathbb{C}$ picks up an id $\mathrm{ID_i}$ from $\mathbb{ID}$ and runs the SignKeyGen(PP, $\mathsf{MK}, \mathsf{ID_i}$) algorithm to get the signing key. Then it runs the $\mathsf{OffSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK_{ID_i}}, \{\mathsf{PK_{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm and the $\mathsf{OnSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK_{ID_i}}, \{\mathsf{PK_{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}}, (\mathbb{M}, \rho_\mathsf{e}), \mathsf{m}, \mathsf{CT}')$ algorithm to return the ciphertext.

4) *De-signcryption:* $\mathbb{A}$ submits a ciphertext CT which involves the associated encryption access structure $(\mathbb{M}, \rho_\mathrm{e})$, a decryption attribute set $\mathrm{U_{uid}}$, and an identity $\mathrm{ID_i}$. $\mathbb{C}$ first runs the $\mathsf{Vry}(\mathsf{CT}, \{\mathsf{PK_{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm to verify the signature. If it passes, $\mathbb{C}$ runs the algorithms OffDsckeyGen(PP,MK), $\mathsf{OnDsckeyGen}(\mathsf{SK}'_\mathsf{d}, \mathsf{U_{uid}})$, and $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$, then runs the algorithms $\mathsf{OutDsc}(\mathsf{CT}, \mathsf{U_{uid}}, \mathsf{SK_{od}})$ and $\mathsf{Designcrypt}(\mathsf{OCT}, \mathsf{CT}, \mathsf{SK_d})$. Finally, $\mathbb{C}$ returns the plaintext m to $\mathbb{A}$.

Challenge: $\mathbb{A}$ submits two messages $m_1^*, m_2^* \in \mathrm{M}$ with equal length and an identity set $\mathbb{ID}^*$. $\mathbb{C}$ selects an identity $\mathrm{ID_i^*}$ from $\mathbb{ID}^*$ and runs the SignKeyGen algorithm to get the signing key. Then, $\mathbb{C}$ randomly selects a bit $b \in \{0, 1\}$ and encrypts the massage m_b^*. $\mathbb{C}$ runs the $\mathsf{OffSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK_{ID_i}}, \{\mathsf{PK_{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ and $\mathsf{OnSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK_{ID_i}}, \{\mathsf{PK_{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}}, (\mathbb{M}, \rho_\mathsf{e}), \mathsf{m_b^*}, \mathsf{CT}')$ algorithms, then returns the ciphertext.

Query II: $\mathbb{A}$ can continue to query as in Query I but it cannot issue the De-signcryption queries of an attribute set $\mathrm{U_{uid}}$ which satisfies the challenge access structure $(\mathbb{M}^*, \rho_e^*)$.

Guess: $\mathbb{A}$ outputs a guess $b' \in \{0, 1\}$. If $b' = b$, $\mathbb{C}$ outputs 1, and 0 otherwise.

The advantage of $\mathbb{A}$ in this game is defined as: $\mathrm{Adv}_{\mathbb{A}}^{\mathrm{IND-sES-CCA2}} = \Pr[b' = b] - \frac{1}{2}$.

Definition 1: An O^3HSC scheme is said to be $(\mathbf{T}, \mathrm{q_{sk}}, \mathrm{q_{dk}}, \mathrm{q_{sc}}, \mathrm{q_{ds}}, \varepsilon)$-IND-sES-CCA2 secure if for any PPT adversary $\mathbb{A}$ running in time at most $\mathbf{T}$ and making at most $\mathrm{q_{sk}}$ Signing Key queries, $\mathrm{q_{dk}}$ De-signcryption Key and Outsourced De-signcryption Key queries, $\mathrm{q_{sc}}$ Signcryption queries and $\mathrm{q_{ds}}$ De-signcryption queries, the advantage defined above is at most ε, where ε is negligible.

2) Ciphertext Unforgeability: The ciphertext existential unforgeability security model of an O^3HSC scheme is defined against adaptive chosen message and identity attack (EUF-CMIA). Details of the game $\mathrm{Game}_{\mathbb{A}}^{\mathrm{EUF-CMIA}}(\kappa)$ between a challenger $\mathbb{C}$ and a PPT adversary $\mathbb{A}$ is presented in the following game.

Set Up: Same as in the above message confidentiality game.

Query: $\mathbb{A}$ issues adaptively a polynomially bounded number of the following queries.

1) *Hash Functions:* $\mathbb{A}$ is allowed to query hash functions $\mathrm{H_2}$ and $\mathrm{H_3}$ for any input.

2) *Signing Key:* $\mathbb{A}$ submits an identity $\mathrm{ID_i}$ to $\mathbb{C}$, then $\mathbb{C}$ runs the $\mathsf{SignKeyGen}$ $(\mathsf{PP}, \mathsf{MK}, \mathsf{ID_i}) \rightarrow (\mathsf{PK}_{\mathsf{ID_i}}, \mathsf{SK}_{\mathsf{ID_i}})$ algorithm and returns $\mathrm{SK_s}$ to $\mathbb{A}$.

3) *De-signcryption Key and Outsourced De-signcryption Key:* $\mathbb{A}$ submits an attribute set $\mathrm{U_{uid}}$. $\mathbb{C}$ sets $\mathbb{S} = \mathbb{S} \cup \mathrm{U_{uid}}$, then it runs the $\mathsf{OffDsckeyGen}$ and $\mathsf{OnDsckeyGen}$ algorithms to generate the designcryption key $\mathrm{SK_d}$. Next, $\mathbb{C}$ first searches the table to answer the outsourced de-signcryption key queries. If this entry exists, it returns $\mathrm{SK_{od}}$ directly. Else, it runs the $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$ algorithm to generate the outsourced de-signcryption key and stores the entry $(\mathrm{U_{uid}}, \mathrm{SK_d}, \mathrm{SK_{od}}, \mathrm{OVK})$ in the table $\mathbb{T}$. Finally, $\mathbb{C}$ returns $(\mathrm{SK_d}, \mathrm{SK_{od}})$ to $\mathbb{A}$.

4) *Signcryption:* $\mathbb{A}$ submits a message $m \in \mathrm{M}$, an identity set $\mathbb{ID}$, and an encryption structure $(\mathbb{M}, \rho_{\mathrm{e}})$. $\mathbb{C}$ picks up an id $\mathrm{ID_i}$ from $\mathbb{ID}$ and runs the $\mathsf{SignKeyGen(PP}$, $\mathsf{MK}, \mathsf{ID_i})$ algorithm to get the signing key. Then it runs the $\mathsf{OffSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK}_{\mathsf{ID_i}}, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm and the $\mathsf{OnSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK}_{\mathsf{ID_i}}, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}}$, $(\mathbb{M}, \rho_{\mathsf{e}}), \mathsf{m_b^*}, \mathsf{CT}')$ algorithm to return the ciphertext.

5) *De-signcryption:* $\mathbb{A}$ submits a ciphertext $\mathbf{CT}$ which involves the associated encryption access structure $(\mathbb{M}, \rho_{\mathrm{e}})$, a decryption attribute set $\mathrm{U_{uid}}$, and an identity $\mathrm{ID_i}$. $\mathbb{C}$ first runs the $\mathsf{Vry}(\mathsf{CT}, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm to verify the signature. If it passes, $\mathbb{C}$ runs the three algorithms $\mathsf{OffDsckeyGen(PP,MK)}$, $\mathsf{OnDsckeyGen}(\mathsf{SK}'_{\mathsf{d}}, \mathsf{U_{uid}})$, and $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$, and then runs the algorithms $\mathsf{OutDsc}(\mathsf{CT}, \mathsf{U_{uid}}, \mathsf{SK_{od}})$ and $\mathsf{Designcrypt}(\mathsf{OCT}, \mathsf{CT}, \mathsf{SK_d})$. Finally, $\mathbb{C}$ returns the plaintext m to $\mathbb{A}$.

Forgery: $\mathbb{A}$ outputs a ciphertext CT^* of a set of identities $\mathbb{ID}^*$ and an access structure $(\mathbb{M}^*, \rho_{\mathrm{e}}^*)$. $\mathbb{A}$ wins this game if (a) $(\mathbb{ID}^*, (\mathbb{M}^*, \rho_{\mathrm{e}}^*))$ has never been queried in the Query phase, (b) no signing key of user in $\mathbb{ID}^*$ was returned during the Signing Key query phase, (c) the $\mathsf{Vry}(\mathsf{CT}^*, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm outputs 1, and (d) $\mathsf{Designcrypt}(\mathsf{OCT}^*, \mathsf{CT}^*, \mathsf{SK_d}) \rightarrow m \neq \perp$. If $\mathbb{A}$ wins, $\mathbb{C}$ outputs 1, and 0 otherwise.

The advantage of $\mathbb{A}$ in this game is defined as: $\mathrm{Adv}_{\mathbb{A}}^{\mathrm{EUF-CMIA}} = \Pr[\mathrm{Game}_{\mathbb{A}}^{\mathrm{EUF-CMIA}}(\kappa) = 1]$.

TABLE II
NOTATIONS

Notation	Meaning
κ	Security Parameter
$\mathbb{G}, \mathbb{G}_T$	Two multiplicative cyclic groups of prime order p
e	An efficient non-degenerate bilinear map
$\mathrm{U} := \{0,1\}^*$	The universal encryption attributes set
$\mathrm{M} := \{0,1\}^{\mathrm{l_m}}$	The message space
$\mathrm{H_1}, \mathrm{H_2}, \mathrm{H_3}, \mathrm{H}'_2$	Four collision resistant hash functions
PP, MK	Public Parameters, Master Key
$\mathrm{ID_i}$	Data Owner Identity
$\mathrm{PK_{ID_i}}, \mathrm{SK_{ID_i}}$	Public and secret identity key pair
$\mathrm{U_{uid}}$	User attribute set
$\mathrm{SK}'_{\mathrm{d}}, \mathrm{SK_d}, \mathrm{SK_{od}}$	Partial, complete and outsourced de-signcryption Key
$\mathbb{ID}$	Identity Set
m	Message
$(\mathbb{M}, \rho_{\mathrm{e}})$	LSSS access matrix and its row label function
$\mathrm{CT}', \mathrm{CT}, \mathrm{OCT}$	Incomplete, complete and out-decrypted ciphertext
$\in_{\mathrm{R}}$	Randomly choose an element from a group

Definition 2: An (O^3HSC) scheme is said to be $(\mathbf{T}, \mathrm{q_{sk}}, \mathrm{q_{dk}}, \mathrm{q_{sc}}, \mathrm{q_{ds}}, \varepsilon)$-EUF-CMIA secure if for any PPT adversary $\mathbb{A}$ running in time at most $\mathbf{T}$ and making at most $\mathrm{q_{sk}}$ Signing Key queries, $\mathrm{q_{dk}}$ De-signcryption Key and Outsourced De-signcryption Key queries, $\mathrm{q_{sc}}$ Signcryption queries and $\mathrm{q_{ds}}$ De-signcryption queries, the advantage defined above is at most ε, where ε is negligible.

IV. Concrete Construction

In this section, we present details of our outsourced online/offline hybrid signcryption (O^3HSC) scheme. Notations used in our scheme are summarized in Table II.

Phase 1 (Set Up): The CA runs the $\mathsf{Setup}(1^{\kappa})$ algorithm to generate the public parameters PP and the master key MK.

Given a security parameter κ, the CA first selects two multiplicative cyclic groups $\mathbb{G}$ and $\mathbb{G}_T$ of prime order p, a generator g of $\mathbb{G}$, and an efficient non-degenerate bilinear map $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$. Let $\mathbb{D} = (p, \mathbb{G}, \mathbb{G}_T, e)$ be the bilinear group description.

Let $\mathrm{U} := \{0,1\}^*$ be the universal encryption attributes set and $\mathrm{M} := \{0,1\}^{l_m}$ be the message space where l_m is the length of messages in our system. We assume that the universal encryption attributes set can be encoded as elements in $\mathbb{Z}_p^*$.

Selects four collision resistant Hash functions $\mathrm{H_1} : \{0,1\}^* \rightarrow \{0,1\}^{l_m}$, $\mathrm{H_2} : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$, $\mathrm{H_3} : \{0,1\}^* \rightarrow \mathbb{G}$ and $\mathrm{H}'_2 : \{0,1\}^* \rightarrow \mathbb{Z}_p^*$. Chooses random exponents $\alpha, \beta \in \mathbb{Z}_p^*$ and computes $Y := e(g,g)^{\alpha}$ and g^{β}. Chooses random elements $\delta_1, \delta_2, \delta_3, h, u, v, w \in \mathbb{G}$. Finally it outputs the master secret key $\mathrm{MK} = [\alpha, \beta]$ and public parameters PP:

$$\mathrm{PP} = \Big[\mathbb{D}, \mathrm{U}, \mathrm{M}, \mathrm{H_1}, \mathrm{H_2}, \mathrm{H_3}, \mathrm{H}'_2, Y, g^{\beta}, g, w, u, h, v, \delta_1, \delta_2, \delta_3\Big].$$

Phase 2 (Key Generation): This phase includes the following two parts.

I. Signing Key Generation: Each patient (data owner) owns an unique identity $\mathrm{ID}_i \in \{0,1\}^*$. The CA generates the signing key pair $(\mathrm{PK_{ID_i}}, \mathrm{SK_{ID_i}})$ for the user's controller to sign with $\mathrm{SK_{ID_i}}$ by running the $\mathsf{SignKeyGen}(\mathsf{PP}, \mathsf{MK}, \mathsf{ID_i})$

algorithm as follows.

$$\mathrm{PK}_{\mathrm{ID_i}} = \mathrm{H}_3(\mathrm{ID_i}) \in \mathbb{G}, \mathrm{SK}_{\mathrm{ID_i}} = \mathrm{PK}_{\mathrm{ID_i}}^{\beta}.$$

II. De-signcryption Key Generation: The CA runs the $\mathsf{DsckeyGen(PP,MK)}$ algorithm to generate the de-signcryption key for each registered user based on their attribute sets, which is divided into an online part and an offline part (the pooling construction was proposed in [8]).

1) *Offline:* The $\mathsf{OffDsckeyGen(PP,MK)}$ algorithm takes as input PP and MK, then it chooses a random element $r \in \mathbb{Z}_p^*$ and computes:

$$K_0' = g^{\alpha} w^{r}, K_r' = g^{r}, K_v = v^{-r}.$$

Then, it chooses two random values $r_x, a_x \in \mathbb{Z}_p^*$ for each attribute in the pool and computes:

$$\left\{K_x = g^{r_x}, K_x' = (u^{a_x} h)^{r_x}\right\}.$$

2) *Online:* The $\mathsf{OnDsckeyGen}(\mathsf{SK}'_{\mathsf{d}}, \mathsf{U_{uid}})$ algorithm takes as input $\mathrm{SK}'_{\mathrm{d}}$ and an user attribute set $\mathrm{U_{uid}} = \{\mathrm{A}_1, \mathrm{A}_2, \ldots, \mathrm{A}_{n_u}\}$ of size n_u. It first chooses an available main component in the pool as $K_0 = g^{\alpha} w^{r}$ and $K_r = g^{r}$. Then it chooses n_u available attribute components in the pool as:

$$K_{x,1} = g^{r_x}, K_{x,2} = K_x' \times K_v = (u^{a_x} h)^{r_x} \cdot v^{-r},$$
$$K_{x,3} = r_x(\mathrm{A_x} - a_x).$$

Finally, it outputs the de-signcryption key $\mathrm{SK_d} = (K_0, K_r, \{K_{x,1}, K_{x,2}, K_{x,3}\}_{n_u})$.

III. Out-De-signcryption Key Generation: To outsource most of the decryption work to a third party, the CA is responsible for generating the outsourced de-signcryption key for users by running the $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$ algorithm. It first chooses a random exponent $z \in \mathbb{Z}_p^*$ to blind the de-signcryption key as: $OK_0 = (K_0)^z, OK_r = (K_r)^z, OK_{x,1} = (K_{x,1})^z$, $OK_{x,2} = (K_{x,2})^z, OK_{x,3} = K_{x,3} \cdot z$. Finally, it outputs the out-designcryption key $\mathrm{SK_{od}} = (OK_0, OK_r, \{OK_{x,1}, OK_{x,2}, OK_{x,3}\}_{n_u})$. After the registration, the CA returns $(\mathrm{SK_{od}}, z)$ to the user.

Phase 3 (Signcryption): This phase contains two parts: offline signcryption and online signcryption where the offline part can be done whenever a mobile device is plugged into a power source. After the message and the encryption access structure are decided, mobile devices (unplugged) can quickly perform the online step and greatly reduce the battery consumption. Our O^3HSC scheme combines the attribute-based encryption and the identity-based ring signature to ensure confidentiality and authenticity of data simultaneously.

1) *Offline:* The $\mathsf{OffSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK_{ID_i}}, \{\mathsf{PK_{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm is done by a controller when it is plugged to a power source, where $\mathbb{ID} = \{\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_{n_s}\}$ is a set of user identities which contains the signer's identity $\mathrm{ID_j}$.

Chooses a random element $s \in \mathbb{Z}_p^*$, and calculates $key = Y^s$ and $C_0 = g^s$. Chooses two random elements $\gamma', \eta \in \mathbb{Z}_p^*$, and calculates $V = (\delta_1^{\gamma'} \delta_2^{\eta} \delta_3)^s$. Denotes by $[key, C_0, V]$ as one main component of ciphertexts. Chooses three random elements $\lambda', x, t \in \mathbb{Z}_p^*$ for each attribute $\mathrm{A_x}$ in the pool and calculates:

$$C_1 = w^{\lambda'} v^{t}, C_2 = (u^{x} h)^{t}, C_3 = g^{t}.$$

Chooses random elements $\xi \in \mathbb{G}$, $g_i' \in \mathbb{G}$ for $i \in \{1, 2, \ldots, n_s\} \backslash \{j\}$. Calculates $g_j' = (\mathrm{PK}_{\mathrm{ID_j}})^{\xi} / \prod_{i=1, i \neq j}^{n_s} g_i'$. Calculates $\sigma' = {\mathrm{SK}_{\mathrm{ID_j}}}^{\xi}$.

2) *Online:* A controller runs the $\mathsf{OnSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK_{ID_i}}, \{\mathsf{PK_{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}}, (\mathbb{M}, \rho_{\mathsf{e}}), \mathsf{m}, \mathsf{CT}')$ algorithm to encrypt and sign the message based on an access structure when it is unplugged, where $(\mathbb{M}, \rho_e)$ is an $l_e \times n_e$ LSSS access matrix with a row labeling function ρ_{e}.

Chooses one available main component from the pool, such as $[key, C_0, V]$. Randomly chooses l_e available attribute components:

$$C_{i,1} = w^{\lambda_i'} v^{t_i}, C_{i,2} = (u^{x_i} h)^{t_i}, C_{i,3} = g^{t_i}.$$

Randomly chooses $y_2, \ldots, y_{n_e} \in \mathbb{Z}_p^*$, sets the vector $\vec{y} = (s, y_2, \ldots, y_{n_e})^{\mathrm{T}}$, and computes a vector of shares of s as $(\lambda_1, \ldots, \lambda_{l_e})^{\mathrm{T}} = \mathbb{M} \cdot \vec{y}$. For $i \in [1, l_e]$, computes:

$$C_{i,4} = \lambda_i - \lambda_i', C_{i,5} = t_i \cdot (\mathrm{A}_{\rho_e(i)} - x_i).$$

Calculates $C = \mathrm{H}_1(key || (\mathbb{M}, \rho_e) || \mathbb{ID}) \oplus m$. Calculates $g_i = g_i'$ and $h_i = \mathrm{H}_2(C || (\mathbb{M}, \rho_e) || \mathbb{ID} || g_i)$ for $i \in \{1, 2, \ldots, n_s\} / \{j\}$. For the signer $\mathrm{ID_j}$, calculates $g_j = g_j' / \prod_{i=1, i \neq j}^{n_s} {\mathrm{PK}_{\mathrm{ID_i}}}^{h_i} = (\mathrm{PK}_{\mathrm{ID_j}})^{\xi} / \prod_{i=1, i \neq j}^{n_s} g_i' \cdot {\mathrm{PK}_{\mathrm{ID_i}}}^{h_i}, h_j = \mathrm{H}_2(C || (\mathbb{M}, \rho_e) || \mathbb{ID} || g_j)$.

Calculates $\sigma = \sigma' \cdot {\mathrm{SK}_{\mathrm{ID_j}}}^{h_j}$. Calculates $\gamma = \mathrm{H}_2'(C_0 || C_{i,1} || C_{i,2} || C_{i,3} || C_{i,4} || C_{i,5} || C || \sigma || (\mathbb{M}, \rho_e) || \mathbb{ID})$ and $V' = s \cdot (\gamma - \gamma')$. Finally, this algorithm outputs the ciphertext as:

$$\mathrm{CT} = \Big(\mathbb{ID}, (\mathbb{M}, \rho_e), C, C_0, \sigma, \{g_i\}_{i \in [1, n_s]}, V, V', \eta,$$
$$\left\{C_{i,1}, C_{i,2}, C_{i,3}, C_{i,4}, C_{i,5}\right\}_{i \in [1, l_e]}\Big).$$

Phase 4 (De-signcryption): This phase contains the following three parts.

1) *Outsourced Decryption:* The user sends its outsourced de-signcryption key $\mathrm{SK_{od}}$ and its attribute set $\mathrm{U_{uid}}$ to the CS. The CS runs the $\mathsf{OutDsc}(\mathsf{CT}, \mathsf{SK_{od}})$ algorithm to partially decrypt for the user.

Decides if the user attribute set $\mathrm{U_{uid}}$ satisfies the access structure $(\mathbb{M}, \rho_{\mathrm{e}})$ contained in the ciphertext. If not, it sends back an error message. Else, it continues to decrypt.

Sets a set $\mathrm{I} = \{i : \rho_e(i) \in \mathrm{U_{uid}}\}$ and finds a set of constants $w_i \in \mathbb{Z}_p^*$ such that $\sum_{i \in I} w_i \cdot \mathbb{M}_i = (1, 0, \ldots, 0)$, where $\mathbb{M}_i$ is the *i*-th row of the access matrix $\mathbb{M}$. Next, it calculates

$$\mathrm{OCT} = \frac{e(C_0, OK_0) e(w^{\sum_{i \in I} C_{i,4} \cdot w_i}, OK_r)^{-1}}{\prod\limits_{i \in I} \Big[e(C_{i,1}, OK_r) \cdot e(C_{i,2} \cdot u^{C_{i,5}}, OK_{\rho_e(i),1})^{-1} \cdot e(C_{i,3}, OK_{\rho_e(i),2} u^{OK_{\rho_e(i),3}}) \Big]^{w_i}}. \tag{1}$$

Sends [OCT, CT] back to the user.

2) *Verification:* Our scheme achieves a public signature verification which means anyone who gets the ciphertext can

perform the $\mathsf{Vry}(\mathsf{CT}, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm to verify the authenticity of data.

Firstly, it calculates $h_i^* = \mathrm{H}_2(C||(\mathbb{M}, \rho_e)||\mathbb{ID}||g_i)$ for $i \in \{1, 2, \ldots, n_s\}$. Then, it verifies if the following equation holds:

$$e\left(g^{\beta}, \prod_{i=1}^{n_s} g_i \cdot \mathrm{PK}_{\mathrm{ID_i}}{}^{h_i^*}\right) = e(g, \sigma). \tag{2}$$

3) *De-signcryption:* After receiving $[\mathrm{OCT}, \mathrm{CT}]$ from the CS, the user runs the $\mathsf{Designcrypt}$ algorithm to recover the data.

Calculates $\gamma^* = \mathrm{H}_2'(C_0||C_{i,1}||C_{i,2}||C_{i,3}||C_{i,4}||C_{i,5}||C|| \sigma||(\mathbb{M}, \rho_e)||\mathbb{ID})$. Verifies if the following equation holds.

$$e\Big(g, V \cdot \delta_1^{V'}\Big) = e\Big(C_0, \delta_1^{\gamma^*} \delta_2^{\eta} \delta_3\Big).$$

If yes, it goes to the next step. Calculates $Y^s = \mathrm{OCT}^{1/z}$ to recover the message as $m = C \oplus \mathrm{H}_1(Y^s||(\mathbb{M}, \rho_e)||\mathbb{ID})$.

V. Security Analysis

A. Complexity Assumptions

Definition 3 [Computational Diffie Hellman Problem (CDH)]: Let $\mathbb{G}$ be a multiplicative cyclic group of order p with a generator g, $g^a \in \mathbb{G}$ and $g^b \in \mathbb{G}$ be two group elements where $a, b \in \mathbb{Z}_p$. The problem of calculating g^{ab} from g^a and g^b is called the Computational Diffie Hellman problem.

Definition 4 [Decisional (q − 1) Problem]: Given a tuple $(\mathbb{D}, g, \{g^{z^i}, g^{k_j}, g^{sk_j}, \; g^{z^i k_j}, g^{z^i/k_j^2}\}_{(i,j)\in[q,q]}, g^s, Z, \{g^{z^i/k_j}\}_{(i,j)\in[2q,q],i\neq q+1}, \{g^{z^i k_j/k_{j'}^2}\}_{(i,j,j')\in[2q,q,q],j\neq j'}, \{g^{sz^i k_j/k_{j'}}, g^{sz^i k_j/k_{j'}^2}\}_{(i,j,j')\in[q,q,q],j\neq j'})$, where $\mathbb{D}$ is the bilinear group description, κ is the security parameter, $q \in poly(\kappa)$, $g \in \mathbb{G}$ is a generator of $\mathbb{G}$ and $s, z, k_1, \ldots, k_q \in_R \mathbb{Z}_p^*$. The problem of determining whether $Z = e(g,g)^{s \cdot z^{q+1}}$ or $Z \in_R \mathbb{G}_T$ is called the decisional $(q - 1)$ problem.

B. IND-CCA2

Theorem 1: Assume that $\mathrm{H}_1, \mathrm{H}_2, \mathrm{H}_3$ and H_2' are collision resistant hash functions. If a PPT adversary $\mathbb{A}$ with a challenge access structure $(\mathbb{M}, \rho_{\mathrm{e}})$ is capable of breaking our scheme's $(\mathbf{T}, q_{sk}, q_{dk}, q_{sc}, q_{ds}, \varepsilon) - \mathrm{IND} - \mathrm{sES} - \mathrm{CCA2}$ secure in the random oracle model, then there is a PPT algorithm that can solve the $(q-1)$ problem with advantage at least $\frac{\varepsilon}{2}(1 - \frac{q_{ds}}{p})$, where $\mathbb{M}$ is an LSSS matrix of size $l_e^* \times n_e^*$, where $l_e^*, n_e^* \leq q$.

Proof: Suppose $\mathbb{A}$ is a PPT adversary that can break our O^3HSC scheme with probability ε, then we can construct an algorithm $\mathbb{B}$ that can solve the $(q - 1)$ problem with probability at least $\frac{\varepsilon}{2}(1 - \frac{q_{ds}}{p})$. Given a $(q - 1)$ problem instance:

$$\Bigg(\mathbb{D}, g, g^s, \left\{g^{z^i}, g^{k_j}, g^{sk_j}, g^{z^i k_j}, g^{z^i/k_j^2}\right\}_{(i,j)\in[q,q]},$$
$$\{g^{z^i/k_j}\}_{(i,j)\in[2q,q],i\neq q+1}, \left\{g^{z^i k_j/k_{j'}^2}\right\}_{(i,j,j')\in[2q,q,q],j\neq j'},$$
$$\left\{g^{sz^i k_j/k_{j'}}, g^{sz^i k_j/k_{j'}^2}\right\}_{(i,j,j')\in[q,q,q],j\neq j'}, Z_\mu\Bigg),$$

where $\mathbb{D} = [p, \mathbb{G}, \mathbb{G}_T, e]$ is the bilinear group description and g is a generator of $\mathbb{G}$. If $\mu = 0$, $Z_\mu = e(g,g)^{s \cdot z^{q+1}}$. Else, Z_μ is a random element of $\mathbb{G}_T$.

Details of the game $\mathrm{Game}_{\mathbb{A}}^{\mathrm{IND-sES-CCA2}}(\kappa)$ between the PPT adversary $\mathbb{A}$ and the algorithm $\mathbb{C}$ is presented as follows where $\mathbb{C}$ plays as the challenger.

1. *Initialization:* $\mathbb{C}$ specifies the attribute universe $\mathrm{U} = \{0,1\}^*$ and the message space $\mathrm{M} = \{0,1\}^{l_m}$. $\mathbb{A}$ submits a challenge access structure $(\mathbb{M}^*, \rho_e^*)$ where $\mathbb{M}^*$ is an $l_e^* \times n_e^*$ LSSS matrix with a row labeling function $\rho_e^* : [l_e^*] \to \mathrm{U}$ and $l_e^*, n_e^* \leq q$. The i-th row of $\mathbb{M}^*$ is denoted by $\mathbb{M}_i^* = (\mathbb{M}_{i,1}^*, \ldots, \mathbb{M}_{i,n_e^*}^*)$.

2. *Set Up:* $\mathbb{C}$ selects randomly $\alpha' \in \mathbb{Z}_p^*$ and sets $Y = e(g,g)^{\alpha'} \cdot e(g^z, g^{z^q})$ by implicitly setting $\alpha = \alpha' + z^{q+1}$. Notice that $\mathbb{C}$ does not know z^{q+1}. $\mathbb{C}$ selects $\beta, u', h', v', v_1, v_2, v_3, \theta_2, \theta_3 \in_R \mathbb{Z}_p^*$ to set $g^\beta, g, w = g^z$. Calculates $u = g^{u'} \prod_{(j,\iota)\in[l_e^*,n_e^*]}(g^{z^\iota/k_j^2})^{\mathbb{M}_{j,\iota}^*}, \delta_1 = g^z g^{v_1}$, $v = g^{v'} \prod_{(j,\iota)\in[l_e^*,n_e^*]}(g^{z^\iota/k_j})^{\mathbb{M}_{j,\iota}^*}, \delta_2 = (g^z)^{\theta_2} g^{v_2}, h = g^{h'} \prod_{(j,\iota)\in[l_e^*,n_e^*]}(g^{z^\iota/k_j^2})^{-\rho_e^*(j)\mathbb{M}_{j,\iota}^*}, \delta_3 = (g^z)^{\theta_3} g^{v_3}$.

$\mathbb{C}$ selects four collision resistant hash functions $\mathrm{H}_1 : \{0,1\}^* \to \{0,1\}^{l_m}$, $\mathrm{H}_2 : \{0,1\}^* \to \mathbb{Z}_p^*$, $\mathrm{H}_3 : \{0,1\}^* \to \mathbb{G}$ and $\mathrm{H}_2' : \{0,1\}^* \to \mathbb{Z}_p^*$.

Finally, $\mathbb{C}$ returns the public parameters $\mathrm{PP} = [\mathbb{D}, \mathrm{U}, \mathrm{M}, \mathrm{H}_1, \mathrm{H}_2, \mathrm{H}_3, \mathrm{H}_2', Y, g^\beta, g, w, u, h, v, \delta_1, \delta_2, \delta_3]$ to $\mathbb{A}$ and keeps the master key $\mathrm{MK} = [\alpha, \beta]$ secret.

3. *Query I:* $\mathbb{A}$ is allowed to issue a polynomially bounded number queries adaptively.

1) *Signing Key:* $\mathbb{C}$ initializes an empty table $\mathbb{T}$. Every time $\mathbb{A}$ submits an identity $\mathrm{ID_i}$, $\mathbb{C}$ searches the table $\mathbb{T}$. If the entry $(\mathrm{ID_i}, *, \mathrm{H}_3(\mathrm{ID_i}), \mathrm{SK}_{\mathrm{ID_i}} = (\mathrm{H}_3(\mathrm{ID_i}))^\beta)$ is available, $\mathbb{C}$ returns the associated $\mathrm{SK_s}$. Else, $\mathbb{C}$ calculates and returns $\mathrm{SK}_{\mathrm{ID_i}} = (\mathrm{H}_3(\mathrm{ID_i}))^\beta)$ to $\mathbb{A}$ while storing it in $\mathbb{T}$.

2) *De-signcryption Key and Outsourced De-signcryption Key:* $\mathbb{A}$ submits an attribute set $\mathrm{U_{uid}}$ which does not satisfy the challenge access structure $(\mathbb{M}^*, \rho_{\mathrm{e}}^*)$. $\mathbb{C}$ sets $\mathbb{S} = \mathbb{S} \cup \mathrm{U_{uid}}$, then it runs the algorithms $\mathsf{OffDsckeyGen(PP{,}MK)}$ and $\mathsf{OnDsckeyGen}(\mathsf{SK}_\mathsf{d}', \mathsf{U_{uid}})$ and generates the de-signcryption key $\mathrm{SK_d}$ as follows.

$\mathbb{C}$ selects a set of numbers $\nu_2, \ldots, \nu_{\mathrm{n_e^*}}$ and sets a vector $\tilde{\nu} = (-1, \nu_2, \ldots, \nu_{\mathrm{n_e^*}})$ such that $\mathbb{M}_{\mathrm{i}}^* \cdot \tilde{\nu} = 0$ for all $\rho_{\mathrm{e}}^*(\mathrm{i}) \in \mathrm{U_{uid}}$. It also selects $r' \in_R \mathbb{Z}_p^*$ and implicitly sets $r = r' + \sum_{i\in[n_e^*]} \nu_i \cdot z^{q+1-i}$. It simulates the de-signcryption key as:

$$K_0 = g^{\alpha'}(g^z)^{r'} \prod_{i=2}^{n_e^*} \Big(g^{z^{q+2-i}}\Big)^{\nu_i},$$
$$K_r = g^{r'} \prod_{i\in[n_e^*]} \Big(g^{z^{q+1-i}}\Big)^{\nu_i}.$$

Then, it picks up $r_x', a_x \in_R \mathbb{Z}_p^*$ for each attribute $\mathrm{A}_x \in \mathrm{U_{uid}}$ and implicitly sets $r_x = r_x' + r \cdot \sum_{\substack{i'\in[l_e^*] \\ \rho_e^*(i')\notin U_{uid}}} \frac{k_{i'}}{x - \rho_e^*(i')}$.

The rest of the de-signcryption key is simulated as:

$$\begin{aligned}
K_{x,1} &= g^{r'_x} \prod_{\substack{i' \in [l_e^*] \\ \rho_e^*(i') \notin \mathrm{U}_{\mathrm{uid}}}} \left(g^{k_{i'}}\right)^{r'/(x-\rho_e^*(i'))} \\
&\quad \cdot \prod_{\substack{(i,i') \in [n_e^*, l_e^*] \\ \rho_e^*(i') \notin \mathrm{U}_{\mathrm{uid}}}} \left(g^{k_{i'} z^{q+1-i}}\right)^{\nu_i/(x-\rho_e^*(i'))}, \\
K_{x,2} &= (u^{a_x} h)^{r'_x} \cdot \left(K'_{x,i}/g^{r'_x}\right)^{a_x u' + h'} \\
&\quad \cdot \prod_{\substack{(i',j,\iota) \in [l_e^*, l_e^*, n_e^*] \\ \rho_e^*(i') \notin \mathrm{U}_{\mathrm{uid}}}} \left(g^{z' k_{i'}/k_j^2}\right)^{r'\left(a_x - \frac{\rho_e^*(j)\mathbb{M}^*_{j,\iota}}{a_x - \rho_e^*(i')}\right)} \\
&\quad \cdot \prod_{\substack{(i,i',j,\iota) \in [n_e^*, l_e^*, l_e^*, n_e^*] \\ \rho_e^*(i') \notin \mathrm{U}_{\mathrm{uid}}, (j \neq i' \vee i \neq \iota)}} \left(g^{k_{i'} z^{q+1-i+\iota}/k_j^2}\right)^{\nu_i\left(a_x - \frac{\rho_e^*(j)\mathbb{M}^*_{j,\iota}}{a_x - \rho_e^*(i')}\right)} \\
&\quad \cdot v^{-r'} \cdot \prod_{i \in [n_e^*]} \left(g^{z^{q+1-i}}\right)^{-v'\nu_i} \\
&\quad \cdot \prod_{\substack{(i,j,\iota) \in [n_e^*, l_e^*, n_e^*] \\ i \neq \iota}} \left(g^{z^{q+1-i+\iota}/k_j}\right)^{-\nu_i \mathbb{M}^*_{j,\iota}}, \\
K_{x,3} &= \left(r'_x + r \sum_{\substack{i' \in [l_e^*] \\ \rho_e^*(i') \notin \mathrm{U}_{\mathrm{uid}}}} \frac{k_{i'}}{x - \rho_e^*(i')} \right) \cdot (\mathrm{A}_x - a_x).
\end{aligned}$$

Next, $\mathbb{C}$ first searches the table to answer the outsourced de-signcryption key queries. If this entry exists, it returns $\mathrm{SK}_{\mathrm{od}}$ directly. Else, it runs the $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$ algorithm to generate the outsourced de-signcryption key and stores the entry $(\mathrm{U}_{\mathrm{uid}}, \mathrm{SK}_{\mathrm{d}}, \mathrm{SK}_{\mathrm{od}}, \mathrm{OVK} = z)$ in the table $\mathbb{T}$. It chooses $z \in \mathbb{Z}_p^*$ to simulate the outsourced de-signcryption key as: $OK_0 = (K_0)^z$, $OK_r = (K_r)^z$, $OK_{x,1} = (K_{x,1})^z$, $OK_{x,2} = (K_{x,2})^z$, $OK_{x,3} = (K_{x,3})^z$.

Finally, $\mathbb{C}$ returns $(\mathrm{SK}_{\mathrm{d}}, \mathrm{SK}_{\mathrm{od}})$ to $\mathbb{A}$. In our simulation, all values of de-signcryption keys and outsourced de-signcryption keys can be simulated by $\mathbb{C}$ with the $(q-1)$ problem instance. From the view of $\mathbb{A}$, all values of keys are identical to that of the original scheme.

3) *Signcryption:* $\mathbb{A}$ submits a message $m \in \mathrm{M}$, a signer id $\mathrm{ID_i}$, and an encryption structure $(\mathbb{M}, \rho_{\mathrm{e}})$. $\mathbb{C}$ runs the $\mathsf{SignKeyGen}(\mathsf{PP}, \mathsf{MK}, \mathsf{ID_i})$ algorithm to get the signing key.

Then it runs the $\mathsf{OffSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK}_{\mathsf{ID_i}}, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ and $\mathsf{OnSigncrypt}(\mathsf{PP}, \mathbb{ID}, \mathsf{SK}_{\mathsf{ID_i}}, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}}, (\mathbb{M}, \rho_{\mathrm{e}}), \mathsf{m}, \mathsf{CT}')$ algorithms, and returns the ciphertext.

4) *De-signcryption:* $\mathbb{A}$ submits a ciphertext CT which involves the associated encryption access structure $(\mathbb{M}, \rho_e)$, a decryption attribute set $\mathrm{U}_{\mathrm{uid}}$, and an identity $\mathrm{ID_i}$. $\mathbb{C}$ first runs the $\mathsf{Vry}(\mathsf{CT}, \{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i} \in \mathbb{ID}})$ algorithm to verify the signature. If it passes and $\mathrm{U}_{\mathrm{uid}}$ does not satisfy the challenge access structure, $\mathbb{C}$ runs the algorithms OffDsckeyGen(PP,MK), $\mathsf{OnDsckeyGen}(\mathsf{SK'_d}, \mathsf{U_{uid}})$ and $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$ to generate the related key. And it runs the $\mathsf{OutDsc}(\mathsf{CT}, \mathsf{U_{uid}}, \mathsf{SK_{od}})$ algorithm and the $\mathsf{Designcrypt}(\mathsf{OCT}, \mathsf{CT}, \mathsf{SK_d})$ algorithm to answer $\mathbb{A}$. Else, $\mathbb{C}$ checks if $\gamma + \eta\theta_2 + \theta_3 = 0$ (the probability of this event is at most $\frac{1}{p}$) where $\gamma = \mathrm{H}'_2(C_0 || C_{i,1} || C_{i,2} || C_{i,3} || C_{i,4} || C_{i,5} || C || \sigma || (\mathbb{M}, \rho_e) || \mathbb{ID})$. If yes, this simulation aborts and $\mathbb{C}$ outputs a random bit. Else, $\mathbb{C}$ calculates $Y^s = e(C_0, g^{\alpha'}) e(V \cdot \delta_1^{V'}/C_0^{\gamma v_1 + \eta v_2 + v_3}, (g^{z^q})^{(\gamma + \eta\theta_2 + \theta_3)^{-1}})$.

Finally, $\mathbb{C}$ returns $m = C \oplus \mathrm{H}_1(Y^s || (\mathbb{M}, \rho_e) || \mathbb{ID})$ to $\mathbb{A}$.

The value of Y simulated by $\mathbb{C}$ is identical to that of the original scheme from the view of $\mathbb{A}$ as:

$$\begin{aligned}
Y^s &= e\left(V \cdot \delta_1^{V'}/C_0^{\gamma v_1 + \eta v_2 + v_3}, \left(g^{z^q}\right)^{(\gamma + \eta\theta_2 + \theta_3)^{-1}}\right) e\left(C_0, g^{\alpha'}\right) \\
&= e\left(\frac{(g^s)^{z\gamma + v_1\gamma + z\theta_2\eta + v_2\eta + z\theta_3 + v_3}}{C_0^{\gamma v_1 + \eta v_2 + v_3}}, \left(g^{z^q}\right)^{(\gamma + \eta\theta_2 + \theta_3)^{-1}}\right) \\
&\quad \cdot e\left(C_0, g^{\alpha'}\right) \\
&= e\left((g^s)^{z\gamma + z\theta_2\eta + z\theta_3}, \left(g^{z^q}\right)^{(\gamma + \eta\theta_2 + \theta_3)^{-1}}\right) e\left(C_0, g^{\alpha'}\right) \\
&= e(g, g)^{s\alpha}.
\end{aligned}$$

4. *Challenge:* $\mathbb{A}$ submits two messages $m_1^*, m_2^* \in \mathrm{M}$ with equal length and an identity set $\mathbb{ID}^*$ of size n_s^*. $\mathbb{C}$ selects an identity $\mathrm{ID}_{\mathrm{j}}^*$ from $\mathbb{ID}^*$ to run the $\mathsf{SignKeyGen}(\mathsf{PP}, \mathsf{MK}, \mathsf{ID}_{\mathsf{j}}^*)$ algorithm and get the signing key.

Then, $\mathbb{C}$ randomly selects a bit $b \in \{0, 1\}$ to encrypt the massage m_b^*. It first chooses $o_2, \ldots, o_{n_e^*} \in_R \mathbb{Z}_p^*$ and sets the vector $\vec{o} = (s, sz + o_2, sz^2 + o_3, \ldots, sz^{n_e^* - 1} + o_{n_e^*})$. We can get that $\lambda_i^* = \mathbb{M}_i^* \cdot \vec{o} = \sum_{j \in [n_e^*]} \mathbb{M}^*_{i,j} sz^{j-1} + \sum_{j=2}^{n_e^*} \mathbb{M}^*_{i,j} o_j$ for $\forall i \in [l_e^*]$. Next, $\mathbb{C}$ chooses $r_{01}, \tilde{\gamma} \in_R \mathbb{Z}_p^*$, $\lambda'_i, x'_i \in_R \mathbb{Z}_p^*$ for $i \in [l_e^*]$. Moreover, it chooses $g_i \in_R \mathbb{G}$ for $i \in \{1, 2, \ldots, n_s^*\}/\{j\}$ and $h'_j, z \in_R \mathbb{Z}_p^*$. It implicitly sets $r' = r_{01} - z^q$ and $t_i = -s \cdot k_i$ to simulate the ciphertext as follows:

$$\begin{aligned}
C_0^* &= g^s, C^* = \mathrm{H}_1\left(Z \cdot e(g^s, g)^{\alpha'} || (\mathbb{M}^*, \rho_e^*) || \mathbb{ID}\right) \oplus m_b^*, \\
C^*_{i,1} &= w^{\sum_{j=2}^{n_e^*} \mathbb{M}^*_{i,j} \cdot o_j} \left(g^{s \cdot k_i}\right)^{-v} w^{-\lambda'_i} \prod_{\substack{(j,\iota) \in [l_e^*, n_e^*] \\ j \neq i}} \left(g^{sz^\iota k_i/k_j}\right)^{-\mathbb{M}^*_{j,\iota}}, \\
h_i^* &= \mathrm{H}_2(C^* || (\mathbb{M}^*, \rho_e^*) || \mathbb{ID}^* || g_i), \eta^* = -(\gamma^* + \theta_3)/\theta_2, \\
C^*_{i,2} &= \left(g^{s \cdot k_i}\right)^{-\left(u'\rho_e^*(i) + h'\right)} u^{-x'_i} \prod_{\substack{(j,\iota) \in [l_e^*, \\ n_e^*] j \neq i}} \left(g^{\frac{sz^\iota k_i}{k_j^2}}\right)^{\left(\rho_e^*(j) - \rho_e^*(i)\right)\mathbb{M}^*_{j,\iota}}, \\
C^*_{i,3} &= \left(g^{s \cdot k_i}\right)^{-1}, C^*_{i,4} = \lambda'_i, C^*_{i,5} = x'_i, \\
g_j^* &= g^z/\mathrm{H}_3\left(\mathrm{ID}_j^*\right)^{h'_j} \cdot \prod_{i=1, i \neq j}^{n_s^*} g_i \mathrm{H}_3(\mathrm{ID}_i)^{h_i}, \\
h_j^* &= \mathrm{H}_2\left(C^* || (\mathbb{M}^*, \rho_e^*) || \mathbb{ID}^* || g_j^*\right), \\
\gamma^* &= \mathrm{H}_2\left(C_0^* || C^*_{i,1} || C^*_{i,2} || C^*_{i,3} || C^*_{i,4} || C^*_{i,5} || C^* || \sigma^* || (\mathbb{M}^*, \rho_e^*) || \mathbb{ID}^*\right), \\
V^* &= (g^s)^{\gamma^* v_1 + \eta^* v_2 + v_3} \delta_1^{-\tilde{\gamma}}, V^{*\prime} = \tilde{\gamma}, \sigma^* = \left(g^\beta\right)^z.
\end{aligned}$$

Finally, $\mathbb{C}$ returns the ciphertext $\mathrm{CT} = ((\mathbb{M}^*, \rho_e^*), \mathbb{ID}^*, C^*, C_0^*, \sigma^*, \{g_i^*\}_{i \in [1, n_s^*]}, \{C^*_{i,1}, C^*_{i,2}, C^*_{i,3}, C^*_{i,4}, C^*_{i,5}\}_{i \in [1, l_s^*]}, V^*, V^{*\prime}, \eta^*)$ to $\mathbb{A}$.

If $Z_\mu = e(g, g)^{s \cdot z^{q+1}}$, the simulated ciphertext is identical to that of the original scheme. Else, Z_μ is a random element of $\mathbb{G}_T$, then the simulated ciphertext is completely random in the view of $\mathbb{A}$.

5. *Query II:* $\mathbb{A}$ can continue to query as in Query I but it cannot issue the De-signcryption queries of any attribute set which satisfies the challenge access structure $(\mathbb{M}^*, \rho_{\mathrm{e}}^*)$.

6. *Guess:* $\mathbb{A}$ outputs a guess bit b'. If any abort happens, $\mathbb{C}$ outputs a random guess of μ. When no abort occurs, if $b' = b$, $\mathbb{C}$ outputs $\mu' = 0$ and $Z_\mu = e(g, g)^{s \cdot z^{q+1}}$. Else it outputs $\mu' = 1$ and Z_μ is a random element of $\mathbb{G}_T$.

Analysis: The probability that this simulation is aborted is $\Pr[\mathrm{ABT}] = \frac{q_{ds}}{p}$ where q_{ds} is the maximum number of De-signcryption queries that $\mathbb{A}$ is allowed to query during this simulation. Notice that the abort event is independent of the value of μ, thus $\Pr[\mu' = \mu|\mathrm{ABT}] = \frac{1}{2}$. $\mathbb{C}$ outputs $\mu' = 1$ if $b' \neq b$, thus $\Pr[\mu' = \mu|\mu = 1 \wedge \overline{\mathrm{ABT}}] = \frac{1}{2}$. If $\mu = 0$, the probability that $\mathbb{A}$ sees a signcryption of m_b^* is ε based on our assumption. Because $\mathbb{C}$ outputs $\mu' = 0$ when $b' = b$, we can get $\Pr[\mu' = \mu|\mu = 0 \wedge \overline{\mathrm{ABT}}] > \varepsilon + \frac{1}{2}$. In a word, $\mathbb{C}$ solves the $(q - 1)$ problem with at least the following advantage:

$$\begin{aligned}
&\mathrm{Adv}_{\mathbb{A}}^{\mathrm{IND-CCA2}} = \Pr[b' = b] - 1/2 \\
&= \Pr[ABT] \cdot Pr[\mu' = \mu|ABT] + Pr[\overline{ABT}] \\
&\cdot \Pr[\mu' = \mu|\overline{ABT}] - 1/2 \\
&= \frac{q_{ds}}{p} \cdot 1/2 + \left(1 - \frac{q_{ds}}{p}\right) \cdot \big(1/2 \cdot \Pr[\mu' = \mu|\mu = 0 \wedge \overline{\mathrm{ABT}}] \\
&\quad +1/2 \cdot \Pr[\mu' = \mu|\mu = 1 \wedge \overline{ABT}]\big) - 1/2 \\
&> \frac{q_{ds}}{p} \cdot \frac{1}{2} + \left(1 - \frac{q_{ds}}{p}\right) \cdot \left(\frac{1}{2} \cdot \left(\varepsilon + \frac{1}{2}\right) + \frac{1}{2} \cdot \frac{1}{2}\right) - \frac{1}{2} \\
&= \frac{\varepsilon}{2}\left(1 - \frac{q_{ds}}{p}\right). \qquad \blacksquare
\end{aligned}$$

C. EUF-CMIA

Theorem 2: O^3HSC scheme is EUF-CMIA secure in the adaptive model under the assumption that the CDH problem is hard.

Proof: Suppose $\mathbb{A}$ is a PPT adversary that can break our O^3HSC scheme with non-negligible probability ε, we can construct an algorithm $\mathbb{C}$ that can solve the CDH problem with probability at least ε. Given a random instance (g, g^a, g^b) of the CDH problem, the aim of $\mathbb{C}$ is to compute g^{ab}. Details of the game $\mathrm{Game}_{\mathbb{A}}^{\mathrm{EUF-CMIA}}(\kappa)$ between the PPT adversary $\mathbb{A}$ and the algorithm $\mathbb{C}$ is presented as follows where $\mathbb{C}$ plays as the challenger.

1. *Set Up:* $\mathbb{C}$ selects randomly $\alpha' \in \mathbb{Z}_p^*$ and sets $Y = e(g, g)^{\alpha'} \cdot e(g^z, g^{z^q})$ by implicitly setting $\alpha = \alpha' + z^{q+1}$. Notice that $\mathbb{C}$ does not know z^{q+1}. And $\mathbb{C}$ selects randomly $\beta \in \mathbb{Z}_{\mathrm{p}}^*$. $\mathbb{C}$ selects randomly $u', h', v', v_1, v_2, v_3 \in \mathbb{Z}_p^*$ and sets $g^\beta = g^\beta, g = g, w = g^z, u = g^{u'}, h = g^{h'}, v = g^{v'}, \delta_1 = g^{v_1}, \delta_2 = g^{v_2}, \delta_3 = g^{v_3}$. $\mathbb{C}$ selects four collision resistant hash functions $\mathrm{H}_1 : \{0,1\}^* \to \{0,1\}^{l_m}$, $\mathrm{H}_2 : \{0,1\}^* \to \mathbb{Z}_p^*$, $\mathrm{H}_3 : \{0,1\}^* \to \mathbb{G}$ and $\mathrm{H}_2' : \{0,1\}^* \to \mathbb{Z}_p^*$ where $\mathrm{H}_2, \mathrm{H}_3$ are modeled as random oracles.

Finally, $\mathbb{C}$ returns the public parameters $\mathrm{PP} = [\mathbb{D}, \mathrm{U}, \mathrm{M}, \mathrm{H}_1, \mathrm{H}_2, \mathrm{H}_3, \mathrm{H}_2', Y, g^\beta, g, w, u, h, v, \delta_1, \delta_2, \delta_3]$ to $\mathbb{A}$ and keeps the master key $\mathrm{MK} = [\alpha, \beta]$ secret.

2. *Query:* $\mathbb{A}$ issues adaptively a polynomially bounded number of the following queries. $\mathbb{C}$ initializes four empty tables $\mathbb{T}_{\mathrm{H}_3}$, $\mathbb{T}_{\mathrm{ID}}$, $\mathbb{T}_{\mathrm{H}_2}$, $\mathbb{T}$ and an empty set $\mathbb{S}$.

1) The hash functions H_2 and H_3. $\mathbb{A}$ is allowed to query the two hash functions for any input.

H_2 Query. For any value v queried by $\mathbb{A}$, $\mathbb{C}$ first checks the table $\mathbb{T}_{\mathrm{H}_2}$. If the entry $(v, \mathrm{H}_2(v))$ exists, it returns the related $\mathrm{H}_2(v)$ directly. Else, $\mathbb{C}$ returns a random value $x_v \in \mathbb{Z}_p^*$ and stores the entry $(v, \mathrm{H}_2(v) = x_v)$ in the table $\mathbb{T}_{H_2}$.

H_3 Query. Each time $\mathbb{A}$ submits an identity $\mathrm{ID_i}$, $\mathbb{C}$ searches the table $\mathbb{T}_{\mathrm{ID}}$. If the entry $(\mathrm{ID_i}, *, \mathrm{H}_3(\mathrm{ID_i}))$ exists, $\mathbb{C}$ returns the associated $\mathrm{H}_3(\mathrm{ID_i})$ to $\mathbb{A}$. Else, $\mathbb{C}$ chooses $y_i \in_R \mathbb{Z}_p^*$ (y_i is never appeared in the table $\mathbb{T}_{\mathrm{H}_3}$) and flips a coin $c \in \{0,1\}$ which yields 0 with probability ζ. If $c = 0$, $\mathbb{C}$ sets $\mathrm{H}_3(\mathrm{ID_i}) = g^{y_i}$. Else, $\mathrm{H}_3(\mathrm{ID_i}) = \mathrm{g}^{\mathrm{a}y_i}$. Finally, $\mathbb{C}$ stores $(\mathrm{ID_i}, y_i, \mathrm{H}_3(\mathrm{ID_i}))$ in the table $\mathbb{T}_{\mathrm{ID}}$ and returns $\mathrm{H}_3(\mathrm{ID_i})$ to $\mathbb{A}$.

2) *Signing Key:* Each time $\mathbb{A}$ submits an identity $\mathrm{ID_i}$. $\mathbb{C}$ simulates the H_3 Query to get the associated $\mathrm{H}_3(\mathrm{ID_i})$. Notice that if $c = 0$, $\mathrm{SK}_{\mathrm{ID_i}} = (\mathrm{H}_3(\mathrm{ID_i}))^\beta = g^{y_i \beta}$. Else $\mathrm{SK}_{\mathrm{ID_i}} = (\mathrm{H}_3(\mathrm{ID_i}))^\beta = g^{y_i a \beta}$ and $\mathbb{C}$ aborts.

3) *De-signcryption Key and Outsourced De-signcryption Key:* $\mathbb{A}$ submits an attribute set $\mathrm{U_{uid}}$. $\mathbb{C}$ sets $\mathbb{S} = \mathbb{S} \cup \mathrm{U_{uid}}$, then it runs the algorithms $\mathsf{OffDsckeyGen(PP,MK)}$ and $\mathsf{OnDsckeyGen}(\mathsf{SK}_\mathsf{d}', \mathsf{U}_\mathsf{uid})$ to generate the de-signcryption key $\mathrm{SK_d}$ as follows.

$\mathbb{C}$ selects $r' \in_R \mathbb{Z}_p^*$ and implicitly sets $r = r' - z^q$. It simulates the de-signcryption key as $K_0 = g^{\alpha'}(g^z)^{r'}$ and $K_r = g^{r'}(g^{z^q})^{-1}$. Then, it picks up $r_x, \mathrm{A}_x' \in_R \mathbb{Z}_p^*$ for each attribute $\mathrm{A}_x \in \mathrm{U_{uid}}$. The rest of the de-signcryption key is simulated as $K_{x,1} = g^{r_x}$, $K_{x,2} = (u^{\mathrm{A}_x'} h)^{r_x} v^{-r'} (g^{z^q})^{v'}$ and $K_{x,3} = r_x(\mathrm{A}_x - \mathrm{A}_x')$.

Next, $\mathbb{C}$ first searches the table to answer the outsourced de-signcryption key queries. If this entry exists, it returns $\mathrm{SK_{od}}$ directly. Else, it runs the $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$ algorithm to generate the outsourced de-signcryption key and stores the entry $(\mathrm{U_{uid}}, \mathrm{SK_d}, \mathrm{SK_{od}}, \mathrm{OVK} = z)$ in the table $\mathbb{T}$. It chooses $z \in \mathbb{Z}_p^*$ to simulate the outsourced de-signcryption key as: $OK_0 = (K_0)^z, OK_r = (K_r)^z, OK_{x,1} = (K_{x,1})^z, OK_{x,2} = (K_{x,2})^z, OK_{x,3} = (K_{x,3})^z$.

Finally, $\mathbb{C}$ returns $(\mathrm{SK_d}, \mathrm{SK_{od}})$ to $\mathbb{A}$.

4) *Signcryption:* $\mathbb{A}$ submits a message $m \in_R \mathrm{M}$, an encryption access structure $(\mathbb{M}, \rho)$, and an identity set $\mathbb{ID}$ of n_s users, where $(\mathbb{M}, \rho_e)$ is an LSSS matrix of size $l_e \times n_e$. $\mathbb{C}$ picks up an id $\mathrm{ID_j}$ from $\mathbb{ID}$ and runs the algorithm $\mathsf{SignKeyGen(PP, MK, ID_j)}$ to get the signing key. Then it simulates the ciphertext as follows.

Selects $s \in_R \mathbb{Z}_p^*$ and sets $C = \mathrm{H}_1(Y^s||(\mathbb{M}, \rho_e)||\mathbb{ID}) \oplus m$ and $C_0 = g^s$. Selects $(y_2, \ldots, y_{n_e} \in \mathbb{Z}_p^*)$ and sets $(\lambda_1, \ldots, \lambda_{l_e}) = (s, y_2, \ldots, y_{n_e}) \cdot \mathbb{M}^{\mathrm{T}}$. For each row $i \in [l_e]$, it also selects $\lambda_i', x_i', t_i \in_R \mathbb{Z}_p^*$ and sets $C_{i,1} = w^{\lambda_i} v^{t_i} \lambda^{\lambda_i'}$, $C_{i,2} = (u^{\rho_e(i)} h)^{t_i} u^{-x_i'}$, $C_{i,3} = g^{t_i}$, $C_{i,4} = \lambda_i'$ and $C_{i,5} = x_i'$.

Selects $g_i \in_R \mathbb{G}$ for $i \in \{1, 2, \ldots, n_s\}/\{j\}$ and calculates $h_i = \mathrm{H}_2(C||(\mathbb{M}, \rho_e)||\mathbb{ID}||g_i)$. Selects randomly $h_j', z \in \mathbb{Z}_p^*$ and calculates the following equations while storing h_j in the table $\mathbb{T}_{\mathrm{H}_2} : g_j = g^z/\mathrm{H}_3(\mathrm{ID_j})^{h_j'} \cdot \prod_{i=1, i \neq j}^{n_s^*} g_i \mathrm{H}_3(\mathrm{ID_i})^{h_i}, h_j = \mathrm{H}_2(C||(\mathbb{M}, \rho_e)||\mathbb{ID}||g_j), \sigma = (g^\beta)^z, \gamma = \mathrm{H}_2'(C_0||C_{i,1}||C_{i,2}||C_{i,3}||C_{i,4}||C_{i,5}||C||\sigma||(\mathbb{M}, \rho_e)||\mathbb{ID})$.

Then, it chooses $\gamma', \eta \in_R \mathbb{Z}_p^*$ and calculates $V = (\delta_1^{\gamma'} \delta_2^\eta \delta_3)^s \delta_1^{-\gamma'}$ and $V' = \gamma'$.

TABLE III
STORAGE OVERHEADS

Scheme	PubParaSize	SignKeySize	DscKeySize	CT Size
[30]	$(\mathrm{N_A}+2)\vert\mathbb{G}\vert+\vert\mathbb{G}_\mathrm{T}\vert$	$2\vert\mathbb{G}\vert$	$(\mathrm{N_{dk}}+2)\vert\mathbb{G}\vert$	$(2\mathrm{N_e}+\mathrm{N_u}+2)\vert\mathbb{G}\vert+\vert\mathbb{G}_\mathrm{T}\vert+\vert\mathbb{Z}_\mathrm{p}^*\vert$
[32]	$10\vert\mathbb{G}\vert+\vert\mathbb{G}_\mathrm{T}\vert$	$(\mathrm{N_{sk}}+2)\vert\mathbb{G}\vert$	$(2\mathrm{N_{dk}}+2)\vert\mathbb{G}\vert$	$(3\mathrm{N_e}+\mathrm{N_s}+3)\vert\mathbb{G}\vert+(2\mathrm{N_e}+3)\vert\mathbb{Z}_\mathrm{p}^*\vert+\mathrm{l_m}$
[33]	$10\vert\mathbb{G}\vert+\vert\mathbb{G}_\mathrm{T}\vert$	$(\mathrm{N_{sk}}+2)\vert\mathbb{G}\vert$	$(\mathrm{N_{dk}}+2)\vert\mathbb{G}\vert$	$(3\mathrm{N_e}+\mathrm{N_s})\vert\mathbb{G}\vert+(2\mathrm{N_e}+3)\vert\mathbb{Z}_\mathrm{p}^*\vert$
[35]	$(2\mathrm{N_A}+2)\vert\mathbb{G}\vert+2\vert\mathbb{G}_\mathrm{T}\vert$	$(\mathrm{N_{sk}}+\mathrm{N_A})\vert\mathbb{G}\vert$	$(\mathrm{N_{dk}}+\mathrm{N_A})\vert\mathbb{G}\vert$	$(\mathrm{N_A}-\mathrm{N_e}+\mathrm{N_v})\vert\mathbb{G}\vert+\vert\mathbb{G}_\mathrm{T}\vert+2\vert\mathbb{Z}_\mathrm{p}^*\vert$
Ours	$9\vert\mathbb{G}\vert+\vert\mathbb{G}_\mathrm{T}\vert$	$2\vert\mathbb{G}\vert$	$(2\mathrm{N_{dk}}+2)\vert\mathbb{G}\vert+\mathrm{N_{dk}}\vert\mathbb{Z}_\mathrm{p}^*\vert$	$(3\mathrm{N_e}+\mathrm{N_s}+3)\vert\mathbb{G}\vert+(2\mathrm{N_e}+2)\vert\mathbb{Z}_\mathrm{p}^*\vert+\mathrm{l_m}$

Abbreviations: $\mathrm{N_A}$, $\mathrm{N_{dk}}$ and $\mathrm{N_{sk}}$: the number of attributes involved in the attribute universe, de-signcryption key and signing key. $\mathrm{N_u}$: the number of users in the signcryption user set. $\mathrm{N_v}$: the maximum number of attributes in an access policy that can be revoked. $\mathrm{N_e}$ and $\mathrm{N_s}$: the number of attributes required in encryption and signing. $\mathrm{l_m}$: the length of message.

TABLE IV
COMPUTATION OVERHEADS

Scheme	Signcrption (Online/Offline)	Verification	Designcryption (Out/User)
[30]	$(3\mathrm{N_e}+\mathrm{N_u}+3)\mathrm{E}+\mathrm{E_T}+(\mathrm{N_u}+1)\mathrm{H}$	$\mathrm{N_u}\mathrm{E}+\mathrm{P}+\mathrm{N_u}\mathrm{H}$	$\mathrm{N_I}\mathrm{E_T}+(2\mathrm{N_I}+1)\mathrm{P}+\mathrm{H}$
[32]	$\mathrm{Off:}\ (5\mathrm{N_e}+4\mathrm{N_s}+6)\mathrm{E}+\mathrm{E_T}+\mathrm{N_s}\mathrm{H}$ $\mathrm{On:}\ 3\mathrm{H}$	$(2\mathrm{N_s}+5)\mathrm{E}+2\mathrm{P}+\mathrm{N_s}\mathrm{H}$	$(2\mathrm{N_I}+4)\mathrm{E}++(2\mathrm{N_I}+1)\mathrm{E_T}+(2\mathrm{N_I}+4)\mathrm{P}$
[33]	$\mathrm{Out:}\ (5\mathrm{N_e}+4\mathrm{N_s}+1)\mathrm{E}+\mathrm{N_s}\mathrm{H}$ $\mathrm{DO:}\ (\mathrm{N_s}+3)\mathrm{E}+\mathrm{E_T}+\mathrm{H}$	$(2\mathrm{N_s}+5)\mathrm{E}+(\mathrm{N_s}+4)\mathrm{P}+\mathrm{E_T}+2\mathrm{H}$	$\mathrm{Out:}\ (6\mathrm{N_I}+4)\mathrm{E}+(2\mathrm{N_I}+3)\mathrm{P}$ $\mathrm{DU:}\ \mathrm{E_T}+\mathrm{H}$
[35]	$(\mathrm{N_A}+\mathrm{N_v}-\mathrm{N_e}+6)\mathrm{E}+(\mathrm{N_s}+1)\mathrm{H}$	$\mathrm{Out:5E+3P}$	$\mathrm{Out:}\ \mathrm{E}+3\mathrm{P}$ $\mathrm{DU:}\ 2\mathrm{E}+\mathrm{H}$
Ours	$\mathrm{Off:}\ (5\mathrm{N_e}+6)\mathrm{E}+\mathrm{E_T}$ $\mathrm{On:}\ \mathrm{N_u}\mathrm{E}+(\mathrm{N_u}+2)\mathrm{H}+\mathrm{Sym}$	$\mathrm{N_u}\mathrm{E}+\mathrm{N_u}\mathrm{H}+2\mathrm{P}$	$\mathrm{Out:}\ (2\mathrm{N_I}+1)\mathrm{E}+(2\mathrm{N_I}+1)\mathrm{E_T}+(3\mathrm{N_I}+2)\mathrm{P}$ $\mathrm{DU:}\ 4\mathrm{E}+2\mathrm{P}+\mathrm{H}+\mathrm{Sym}$

Abbreviations: $\mathrm{N_I}$: the number of attributes required in decryption. $\mathrm{E}/\mathrm{E_T}$: one exponentiation operation on group $\mathbb{G}$ / $\mathbb{G}_T$. P: one bilinear pairing operation. H: one hash operation. Sym: operations required in the sym-encryption or the sym-decryption.

Finally, it returns the ciphertext to $\mathbb{A}$.

5) *Designcryption:* $\mathbb{A}$ submits a ciphertext CT which involves the associated encryption access structure $(\mathbb{M},\rho_\mathrm{e})$, a decryption attribute set $\mathrm{U_{uid}}$, and an identity $\mathrm{ID_i}$. $\mathbb{C}$ first runs the algorithm $\mathsf{Vry}(\mathsf{CT},\{\mathsf{PK}_{\mathsf{ID_i}}\}_{\mathsf{ID_i}\in\mathbb{ID}})$ to verify the signature. If it passes, $\mathbb{C}$ runs OffDsckeyGen(PP,MK), $\mathsf{OnDsckeyGen}(\mathsf{SK}'_\mathsf{d},\mathsf{U_{uid}})$ and $\mathsf{OutDsckeyGen}(\mathsf{SK_d})$ to generate the related key. And then it runs $\mathsf{OutDsc}(\mathsf{CT},\mathsf{U_{uid}},\mathsf{SK_{od}})$ and $\mathsf{Designcrypt}(\mathsf{OCT},\mathsf{CT},\mathsf{SK_d})$ to answer $\mathbb{A}$.

3. *Forgery:* $\mathbb{A}$ outputs CT^* of a set of identities $\mathbb{ID}^*$ and an access structure $(\mathbb{M}^*,\rho_\mathrm{e}^*)$. $\mathbb{A}$ wins this game if (1) $(\mathbb{ID}^*,(\mathbb{M}^*,\rho_\mathrm{e}^*))$ has never been queried in the Query phase, (2) no signing key of user in $\mathbb{ID}^*$ was returned during the Signing Key query phase, (3) the $\mathsf{Vry}(\mathsf{CT}^*,\mathsf{PK_s})$ algorithm outputs 1 and (4) $\mathsf{Designcrypt}(\mathsf{OCT},\mathsf{CT},\mathsf{SK_d})\rightarrow m\ \neq\perp$. If $\mathbb{A}$ wins, $\mathbb{C}$ can solve the CDH problem as follows.

Assume that there are two signatures, denoted by $(\{g_i\}_{i\in[n_s]},\sigma_1)$ and $(\{g_i\}_{i\in[n_s]},\sigma_2)$, which are signed by the same identity set. For all $i\in\{1,2,\ldots,n_s\}/\{j\}$, we can easily get $h_i=h'_i$. Then $\mathbb{C}$ can calculate the following equation as the result of the CDH problem:

$$\left(\frac{\sigma_1}{\sigma_2}\right)^{\frac{1}{y_j\left(h_j-h'_j\right)}}=\left(\frac{g^{\beta z_1}}{g^{\beta z_2}}\right)^{\frac{1}{y_j\left(h_j-h'_j\right)}}=\left(\frac{g^{z_1}}{g^{z_2}}\right)^{\frac{\beta}{y_j\left(h_j-h'_j\right)}}$$

$$=\left(\frac{g_j\cdot\mathrm{H}_3\left(\mathrm{ID_j}\right)^{h_j}/\prod_{i=1,i\neq j}^{n_s^*}g_i\mathrm{H}_3(\mathrm{ID_i})^{h_i}}{g_j\cdot\mathrm{H}_3\left(\mathrm{ID_j}\right)^{h'_j}/\prod_{i=1,i\neq j}^{n_s^*}g_i\mathrm{H}_3(\mathrm{ID_i})^{h'_i}}\right)^{\frac{\beta}{y_j\left(h_j-h'_j\right)}}$$

$$=\left(\frac{\mathrm{H}_3\left(\mathrm{ID_j}\right)^{h_j}}{\mathrm{H}_3\left(\mathrm{ID_j}\right)^{h'_j}}\right)^{\frac{\beta}{y_j\left(h_j-h'_j\right)}}=\left(\frac{g^{y_j a h_j}}{g^{y_j a h'_j}}\right)^{\frac{\beta}{y_j\left(h_j-h'_j\right)}}$$

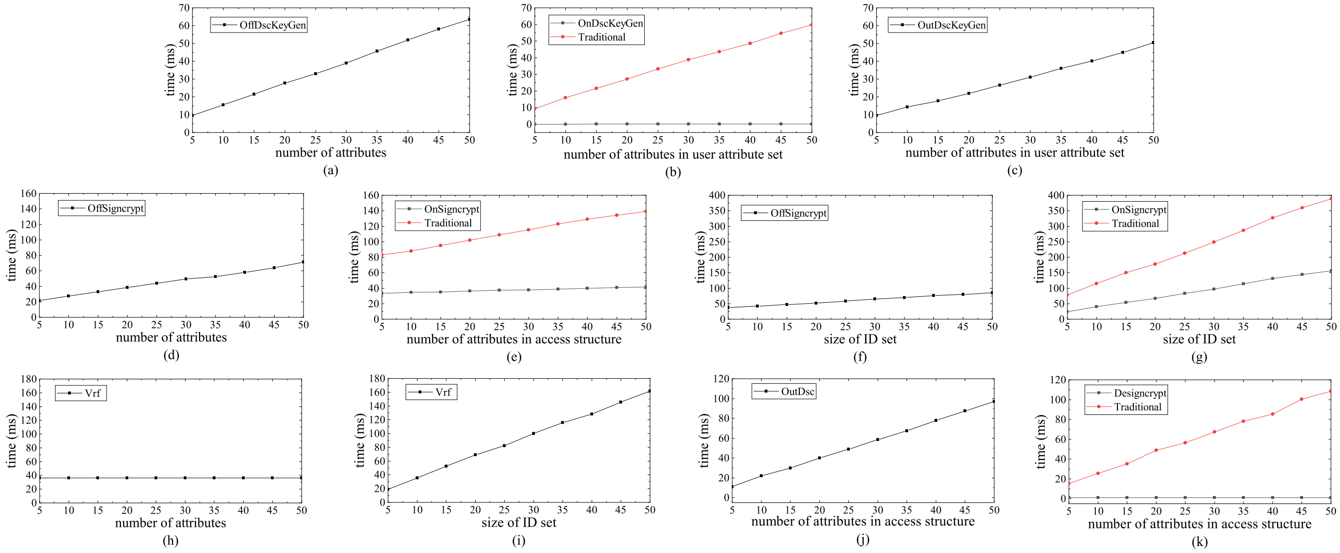
$$=g^{y_j a\left(h_j-h'_j\right)\frac{\beta}{y_j\left(h_j-h'_j\right)}}=g^{a\beta}. \qquad \blacksquare$$

VI. Performance Analysis

This section contains the theoretical analysis of the computational and storage costs of our scheme, as well as the simulation results and analysis of the time costs of several major algorithms. Table III and Table IV summarize the cost comparison between our scheme and several related schemes.

Focus on the storage performance, our O^3HSC scheme and the schemes in [32], [33] have constant-size public parameters while the size of public parameters of the schemes in [30], [35] depends on the size of attribute universe. The signing key of the two identity-based signature schemes (O^3HSC and [30]) is of length two elements in $\mathbb{G}$ while the size of other attribute-based signature schemes depends on the number of attributes in the users' signing attribute sets. Regarding the size of the de-signcryption key, all the schemes listed in Table III are related to the number of attributes in the users' encryption attribute sets. The ciphertext size of all the schemes mentioned in Table III except the one in [35] whose ciphertext size is related to the attribute universe, increases linearly in the number of attributes required for signcryption.

From Table IV, we can see that the schemes in [32] and ours divide the signcryption into an online phase and an offline phase. Thus, the online cost of those two schemes are independent of the number of attributes involved in encryption or signing. The scheme in [33] obtains this result in a different setting: it utilizes the outsourcing signcryption technology. Moreover, the scheme in [33] also outsources its verification to decrease the device overhead. In terms of the cost of de-signcryption, all the schemes in Table IV except these in [30] and [32] adopt the outsourced de-signcryption. It is easy to see that the pairing operations in the four out-designcrypt schemes are independent of the number of attributes required in decryption. Specifically, our scheme requires only four exponentiations in $\mathbb{G}$, two pairings and one hash operations to recover the plaintext.



(a) (b) (c) (d) (e) (f) (g) (h) (i) (j) (k)

Fig. 4. Simulation Results.

The simulation experiments were performed in a Ubuntu-20.04.2.0-desktop system with a 1.10 GHz Intel Core i5-1035G4 CPU and 8 GB RAM. All programs are developed using Charm (version 0.50) [37], which is a Python-based rapid prototyping framework for cryptographic schemes. In order to ensure the reliability of the results, all data are the average of 100 experiments under the same environment.

The computational overhead of the de-signcryption key generation algorithms, including the offline part, online part and the outsourced key generation, is illustrated in Fig. 4 (a-c), where the ordinate is the time cost, and the abscissa in (a), (b) and (c) is the number of attributes in the attribute universe and user attribute set. From Fig. 4, it is easy to see that the time cost of the offline part, the traditional key generation method (all work is done online), and the outsourced key generation are all linear in the number of attributes, while the cost of the online part of the key generation in our scheme is very small, only 0.13ms with 50 attributes. On the contrary, the traditional method costs 60.54ms to generate a de-signcryption key with 50 attributes, which is too heavy for the key generation authority to handle millions of devices in IoT with acceptable latency.

The time cost of signcryption is related to two variables: the number of attributes and the size of the identity set. In our experiment, when the number of attributes varies, the size of the identity set is fixed at 10, otherwise the number of attributes is fixed at 25. As shown in Fig. 4 (d)(e), the time overhead of the offline part of signcryption and the traditional method (all work is done online) grows linearly in the number of attributes, reaching respectively about 71.49ms and 140.68ms with 50 attributes. Although the online signcryption overhead of our scheme is also related to the number of attributes, it has achieved a huge reduction, say only 40.25ms in the case of 50 attributes, which is very practical for mobile device applications. On the other hand, the three time-expenditures also vary with the size of the user identity set, as shown in Fig. 4 (f), (g). In particular, when the size of the identity set is 50, the time for online signcryption takes at most 153.04ms, while the traditional method takes 389.77ms.

Fig. 4 (h-k) illustrates the simulation results of the verification and the de-signcryption algorithms. As we mentioned before, our scheme achieves the public verification. Fig. 4 (h), (i) shows that although the verification time is independent of the number of attributes, it grows linearly in the size of the identity set. Fig. 4 (j), (k) declares that the time overhead of the outsourced de-signcryption and the traditional de-signcryption method (non-outsourced de-signcryption) increases linearly in the number of attributes, reaching 97.21ms and 109.8ms, respectively. However, the user de-signcryption time is irrelevant to the number of attributes in the access structure (only about 1.33 ms) as our scheme outsources most exponentiation and pairing operations to the cloud.

To sum up, under the premise of ensuring safety, our O^3HSC scheme has made great efficiency improvements in key generation, signcryption and de-signcryption through the online-and-offline and outsourcing technologies. Therefore, it is more suitable for mobile devices with limited power supply in environments such as WBAN.

VII. Conclusion

In this paper, we combined the identity-based ring signature and the attribute-based online/offline encryption into one primitive and proposed an outsourced online/offline hybrid signcryption scheme (O^3HSC). On the one hand, we divided the signcryption algorithm into offline and online to meet the needs of mobile devices in WBAN. Moreover, the key generation is divided into offline and online to solve the bottleneck of key issuance in an IoT system with tons of users. On the other hand, we outsourced most of the de-signcryption work to the cloud server, and the users do not need any pairing operations to decrypt. In addition, our signature scheme satisfies public verification. The security analysis shows that our scheme achieves the indistinguishability of ciphertexts under adaptive CCA and the existential unforgeability of signature under adaptive CMIA. The storage and computational performance indicate that our O^3HSC scheme can greatly reduce the computational and storage overload of devices.

Nevertheless, the security and outsourced decryption of the proposed scheme relies on two central and trusted servers. Therefore, in the future, we plan to apply the O^3HSC scheme to more complex environments, such as multi-layers industrial IoT, and implement the edge server to break out the over-centered power of the cloud.

References

[1] S. Chaudhary, A. Singh, and K. Chatterjee, "Wireless body sensor network (WBSN) security and privacy issues: A survey," *Int. J. Comput. Intell. IoT*, vol. 2, no. 2, p. 7, 2019.

[2] A. Sahai and B. Waters, "Fuzzy identity-based encryption," in *Proc. Annu. Int. Conf. Theory Appl. Cryptograph. Techn.*, 2005, pp. 457–473.

[3] Y. Rouselakis and B. Waters, "Practical constructions and new proof methods for large universe attribute-based encryption," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, 2013, pp. 463–474.

[4] B. Waters, "Ciphertext-policy attribute-based encryption: An expressive, efficient, and provably secure realization," in *Proc. Int. Workshop Public Key Cryptogr.*, 2011, pp. 53–70.

[5] X. Liu, H. Zhu, J. Ma, J. Ma, and S. Ma, "Key-policy weighted attribute based encryption for fine-grained access control," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC)*, Sydney, NSW, Australia, 2014, pp. 694–699.

[6] W. Ren, Y. Sun, H. Luo, and M. Guizani, "SILedger: A blockchain and ABE-based access control for applications in SDN-IoT networks," *IEEE Trans. Netw. Service Manag.*, vol. 18, no. 4, pp. 4406–4419, Dec. 2021.

[7] F. Li, M. K. Khan, K. Alghathbar, and T. Takagi, "Identity-based online/offline signcryption for low power devices," *J. Netw. Comput. Appl.*, vol. 35, no. 1, pp. 340–347, 2012.

[8] S. Hohenberger and B. Waters, "Online/offline attribute-based encryption," in *Proc. Int. Workshop Public Key Cryptogr.*, 2014, pp. 293–310.

[9] M. Green, S. Hohenberger, and B. Waters, "Outsourcing the decryption of ABE ciphertexts," in *Proc. 20th USENIX Security Symp. (USENIX Security*, 2011, p. 34.

[10] J. Lai, R. H. Deng, C. Guan, and J. Weng, "Attribute-based encryption with verifiable outsourced decryption," *IEEE Trans. Inf. Forensics Security*, vol. 8, pp. 1343–1354, 2013.

[11] J. Ning, Z. Cao, X. Dong, K. Liang, H. Ma, and L. Wei, "Auditable σ-time outsourced attribute-based encryption for access control in cloud computing," *IEEE Trans. Inf. Forensics Security*, vol. 13, pp. 94–105, 2018.

[12] R. L. Rivest, A. Shamir, and Y. Tauman, "How to leak a secret," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Security*, 2001, pp. 552–565.

[13] Y. Zheng, "Digital signcryption or how to achieve cost(signature & encryption)$\ll$ cost (signature)+ cost (encryption)," in *Proc. Annu. Int. Cryptol. Conf.*, 1997, pp. 165–179.

[14] J. Malone-Lee, "Identity-based signcryption," IACR Cryptol. ePrint Arch., Lyon, France, Rep. 2002/98, 2002.

[15] J. Liu, X. Huang, and J. K. Liu, "Secure sharing of personal health records in cloud computing: Ciphertext-policy attribute-based signcryption," *Future Gener. Comput. Syst.*, vol. 52, pp. 67–76, Nov. 2015.

[16] F. Zhang and K. Kim, "ID-based blind signature and ring signature from pairings," in *Proc. Int. Conf. Theory Appl. Cryptol. Inf. Security*, 2002, pp. 533–547.

[17] S. S. M. Chow, S.-M. Yiu, and L. C. K. Hui, "Efficient identity based ring signature," in *Proc. Int. Conf. Appl. Cryptogr. Netw. Security*, 2005, pp. 499–512.

[18] S. Liu, J. Yu, and L. Chen, "Rewarding and efficient data sharing in EHR system with coalition blockchain assistance," in *Proc. Int. Conf. Wireless Algorithms Syst. Appl.*, 2021, pp. 95–107.

[19] K. Sowjanya and M. Dasgupta, "A ciphertext-policy attribute based encryption scheme for wireless body area networks based on ECC," *J. Inf. Security Appl.*, vol. 54, Oct. 2020, Art. no. 102559.

[20] J. Cui, H. Zhou, Y. Xu, and H. Zhong, "OOABKS: Online/offline attribute-based encryption for keyword search in mobile cloud," *Inf. Sci.*, vol. 489, pp. 63–77, Jul. 2019.

[21] Q. Xu, C. Tan, W. Zhu, Y. Xiao, Z. Fan, and F. Cheng, "Decentralized attribute-based conjunctive keyword search scheme with online/offline encryption and outsource decryption for cloud computing," *Future Gener. Comput. Syst.*, vol. 97, pp. 306–326, Aug. 2019.

[22] B. Qin, R. H. Deng, S. Liu, and S. Ma, "Attribute-based encryption with efficient verifiable outsourced decryption," *IEEE Trans. Inf. Forensics Security*, vol. 10, pp. 1384–1393, 2015.

[23] S. Liu, J. Yu, C. Hu, and M. Li, "Outsourced multi-authority ABE with white-box traceability for cloud-IoT," in *Proc. Int. Conf. Wireless Algorithms Syst. Appl.*, 2020, pp. 322–332.

[24] S. Liu, J. Yu, Y. Xiao, Z. Wan, S. Wang, and B. Yan, "BC-SABE: Blockchain-aided searchable attribute-based encryption for cloud-IoT," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7851–7867, Sep. 2020.

[25] C. Feng, K. Yu, M. Aloqaily, M. Alazab, Z. Lv, and S. Mumtaz, "Attribute-based encryption with parallel outsourced decryption for edge intelligent IoV," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 13784–13795, Nov. 2020.

[26] J. Baek, R. Steinfeld, and Y. Zheng, "Formal proofs for the security of signcryption," in *Proc. Int. Workshop Public Key Cryptogr.*, 2002, pp. 80–98.

[27] X. Huang, W. Susilo, Y. Mu, and F. Zhang, "Identity-based ring signcryption schemes: Cryptographic primitives for preserving privacy and authenticity in the ubiquitous world," in *Proc. 19th Int. Conf. Adv. Inf. Netw. Appl. (AINA) Vol. 1 (AINA Papers)*, vol. 2. Taipei, Taiwan, 2005, pp. 649–654.

[28] M. Gagné, S. Narayan, and R. Safavi-Naini, "Threshold attribute-based signcryption," in *Proc. Int. Conf. Security Cryptography Netw.*, 2010, pp. 154–171.

[29] G. Yu and Z. Cao, "Attribute-based signcryption with hybrid access policy," *Peer-to-Peer Netw. Appl.*, vol. 10, no. 1, pp. 253–261, 2017.

[30] C. Wang, X. Xu, Y. Li, and D. Shi, "Integrating ciphertext-policy attribute-based encryption with identity-based ring signature to enhance security and privacy in wireless body area networks," in *Proc. Int. Conf. Inf. Security Cryptol.*, 2014, pp. 424–442.

[31] N. Eltayieb, R. Elhabob, A. Hassan, and F. Li, "A blockchain-based attribute-based signcryption scheme to secure data sharing in the cloud," *J. Syst. Archit.*, vol. 102, Jan. 2020, Art. no. 101653.

[32] Y. S. Rao, "Attribute-based online/offline signcryption scheme," *Int. J. Commun. Syst.*, vol. 30, no. 16, p. e3322, 2017.

[33] K. P. Kibiwott, Y. N. Zhao, J. Kogo, and F. L. Zhang, "Verifiable fully outsourced attribute-based signcryption system for IoT eHealth big data in cloud computing," *Math. Biosci. Eng.*, vol. 16, no. 5, pp. 3561–3594, 2019.

[34] J. Yu, S. Liu, S. Wang, Y. Xiao, and B. Yan, "LH-ABSC: A lightweight hybrid attribute-based signcryption scheme for cloud-fog-assisted IoT," *IEEE Internet Things J.*, vol. 7, no. 9, pp. 7949–7966, Sep. 2020.

[35] S. Belguith, N. Kaaniche, M. Hammoudeh, and T. Dargahi, "PROUD: Verifiable privacy-preserving outsourced attribute based signcryption supporting access policy update for cloud assisted IoT applications," *Future Gener. Comput. Syst.*, vol. 111, pp. 899–918, Oct. 2020.

[36] X. Liu, Z. Wang, Y. Ye, and F. Li, "An efficient and practical certificateless signcryption scheme for wireless body area networks," *Comput. Commun.*, vol. 162, pp. 169–178, Oct. 2020.

[37] J. A. Akinyele *et al.*, "Charm: A framework for rapidly prototyping cryptosystems," *J. Cryptograph. Eng.*, vol. 3, no. 2, pp. 111–128, 2013.

Suhui Liu received the B.S. and M.S. degrees in computer science from Qufu Normal University, Shandong, China, in 2018 and 2021, respectively. She is currently pursuing the Ph.D. degree with the School of Cyber Science and Engineering, Southeast University, Jiangsu, China. Her main research interests include cloud-assisted IoT data security, functional cryptography, and blockchain technology.

Liquan Chen (Senior Member, IEEE) received the Ph.D. degree from Southeast University, Nanjing, China, in 2005, where he worked as a Postdoctoral Fellow from 2005 to 2007 and an Associate Professor from 2008 to 2018. He worked as a Visiting Scholar with the National University of Singapore, Singapore, from 2011 to 2012. He is currently a Professor with the School of Cyber Science and Engineering, Southeast University. His research interests include information security, cryptography, and network security protocol.

Huaqun Wang received the B.S. degree in mathematics education from Shandong Normal University, China, in 1997, the M.S. degree in applied mathematics from East China Normal University, China, in 2000, and the Ph.D. degree in cryptography from the Nanjing University of Posts and Telecommunications in 2006, where he is currently a Professor. His research interests include applied cryptography, blockchain, network security, and cloud computing security.

Shihui Fu received the Ph.D. degree in mathematics from the Academy of Mathematics and Systems Science, CAS in 2018. He is currently a Postdoctoral Fellow with TU Delft. His current main research interests include cryptographic protocol and zero-knowledge proof.

Lin Shi received the B.S. degree in computer science from Southeast University, China, in 2006, and the M.S. degree in software engineering from the University of Electronic Science and Technology, China, in 2010. He is currently pursuing the Ph.D. degree with the School of Cyber Science and Engineering, Southeast University. His research interests include computational intelligence, information security, and big-data analysis.